

NLP4MusA 2020

**Proceedings of the 1st Workshop on
NLP for Music and Audio**

16–17 October, 2020

Online

©2020 The Association for Computational Linguistics

Order copies of this and other ACL proceedings from:

Association for Computational Linguistics (ACL)
209 N. Eighth Street
Stroudsburg, PA 18360
USA
Tel: +1-570-476-8006
Fax: +1-570-476-0860
acl@aclweb.org

Introduction

Welcome to the 1st Workshop on NLP for Music and Audio. The aim of NLP4MusA is to bring together researchers from various disciplines related to music and audio content, on one hand, and NLP on the other. It embraces the following topics.

- NLP architectures applied to music analysis and generation
- Lyrics analysis and generation
- Exploiting music related texts in music recommendation
- Taxonomy learning
- Podcasts recommendations
- Music captioning
- Multimodal representations

The workshop spans one day split into two days to accommodate an online format while preserving a timezone friendly schedule, which features both live and asynchronous presentations and Q/A sessions. The main topics covered in the accepted papers

The talks of our keynote speakers highlight topics of high relevance in the intersection between music, audio and NLP. The presentation by Colin Raffel discusses what Music Information Retrieval (MIR) can learn from recent transfer learning advances in NLP. Sam Mehr focuses in his talk on the notion of universality in music. NLP4MusA also features an impressive number of industry-led talks by Tao Ye, Fabien Gouyon, Elena Epure, Marion Baranes, Romain Henequin, Sravana Reddy, Rosa Stern, Alice Coucke, Isaac Julien and Shuo Zhang. We include the abstract of their talks in this volume.

In total, we accepted 16 long papers (53% of submissions), following the recommendations of our peer reviewers. Each paper was reviewed by three experts. We are extremely grateful to the Programme Committee members for their detailed and helpful reviews.

Sergio Oramas, Luis Espinosa-Anke, Elena Epure, Rosie Jones, Mohamed Sordo, Massimo Quadrana and Kento Watanabe

October 2020

Organisers:

Sergio Oramas (Pandora)
Luis Espinosa-Anke (Cardiff University)
Elena Epure (Deezer)
Rosie Jones (Spotify)
Mohamed Sordo (Pandora)
Massimo Quadrana (Pandora)
Kento Watanabe (AIST)

Program Committee:

José Camacho-Collados (Cardiff University)
Francesco Barbieri (Snap)
Andrés Ferraro (UPF)
Minz Won (UPF)
Lorenzo Porcaro (UPF)
Albin Correya (UPF)
Pablo Accuosto (UPF)
Morteza Behrooz (Facebook)
Christos Christodouloupoulos (Amazon)
Mark Levy (Apple)
Fabien Guyon (Pandora)
Scott Waterman (Pandora)
Gregory Finley (Pandora)
Zahra Rahimi (Pandora)
Ann Clifton (Spotify)
Sravana Reddy (Spotify)
Aasish Pappu (Spotify)
Manuel Moussallam (Deezer)
Marion Baranes (Deezer)
Romain Hennequin (Deezer)
Horacio Saggion (UPF)
Iñigo Casanueva (Poly AI)
Giuseppe Rizzo (LINKS Foundation)
Pasquale Lisena (EURECOM)
Masataka Goto (AIST)
Peter Knees (TU Wien)
Shuo Zhang (Bose Corporation)
Markus Schedl (TU Linz)
Ichiro Fujinaga (McGill University)
Elena Cabrio (Université Cote d'Azur)
Michael Fell (Université Cote d'Azur)

Richard Sutcliffe (University of Essex)

Invited Speakers:

Colin Raffel, University of Carolina, Chapel Hill & Google Brain

Sam Mehr, Harvard Music Lab

Tao Ye, Amazon

Fabien Gouyon, Pandora/SiriusXM

Elena Epure, Deezer

Marion Baranes, Deezer

Romain Henequin, Deezer

Sravana Reddy, Spotify

Rosa Stern, Sonos

Alice Coucke, Sonos

Isaac Julien, Bose

Shuo Zhang, Bose

Invited Talks

Colin Raffel: What can MIR learn from transfer learning in NLP?

Transfer learning has become the de facto pipeline for natural language processing (NLP) tasks. The typical transfer learning recipe trains a model on a large corpus of unstructured, unlabeled text data using a self-supervised objective and then fine-tunes the model on a downstream task of interest. This recipe dramatically mitigates the need for labeled data and has led to incredible progress on many benchmarks that had previously been far out of reach. In this talk, I'll first give an overview of transfer learning for NLP from the lens of our recent empirical survey. Then, I will argue that transfer learning is massively underutilized in the field of music information retrieval (MIR), particularly in light of the scarcity of labeled music data. To prompt future research, I'll highlight some successful applications of transfer learning in MIR and discuss my own work on creating a large, weakly-labeled music dataset.

Sam Mehr: Universality and diversity in human song

What is universal about music, and what varies? In this talk I will present some highlights from analysis of the Natural History of Song Discography, which includes audio recordings from 86 human societies, to uncover what makes music sound the way it does around the world. Using data from music information retrieval, amateur and expert listener ratings, and manual transcriptions, we find that acoustic features of songs predict their primary behavioral context; that tonality is widespread, perhaps universal; that music varies in rhythmic and melodic complexity; and that elements of melodies and rhythms found worldwide follow power laws. The findings demonstrate basic facts of the human psychology of music that may inform our understanding of aesthetics and our preferences for music.

Tao Ye: Inside a real world conversational music recommender

When a user asks Alexa "Help me find music", there are in fact a multitude of interesting problems to be solved, in the cross-section of Natural Language Understanding, recommendation systems, and advanced natural language generation. In Natural Language Understanding, we encounter intent identification, slots filling, and particular challenges of spoken language understanding (SLU) in the music domain. This is also different from a one-shot command SLU, where users tend to give a clear "play XYZ" intent. In a dialog, users increase variation in their speech and often answer a question casually such as "whatever, I don't care". We rely on both grammatically rules and statistical models to set intent, triggers and fill slots. Machine learning is also applies directly to construct an interactive recommender that makes recommendations more relevant. With real time user critique and feedback, we need to integrate long term user preferences and immediate user requests. Finally, how Alexa speaks to the users also makes a difference in the experience. We tackle the tough problem of making the entire conversation sound natural rather than robotic. Particularly, emotional and empathic tagged speech are used. The challenge is to know when to use these tags to vary speech.

Fabien Gouyon: Lean-back or Lean-in?

In this talk I will go over some of Pandora's latest research and product developments in the realm of voice interactions. I will address how NLU powers unique music listening experiences in the Pandora app, and highlight exciting opportunities for further research and development.

Elena Epure, Marion Baranes & Romain Henequin: “Je ne parle pas anglais””, dealing with multilingualism in MIR

Deezer is a local player, on a global scale. Our goal is to serve a very diverse audience providing a seamless experience worldwide. Consequently, dealing with multilingualism, and more generally with multiculturalism is essential to us. In this talk, we address two topics for which the generalisation to multilingual data and users is particularly important: the user-app interaction through the search engine and the catalogue annotation with multilingual metadata. We conclude by contemplating the state of multilingualism in the music information retrieval (MIR) community.

Sravana Reddy: The Spotify Podcasts Dataset

We present the Spotify Podcasts Dataset, a set of approximately 100K podcast episodes comprised of raw audio files along with accompanying ASR transcripts, that we released for the TREC 2020 Challenge. We will talk about some of the characteristics of this dataset, and our experiments running baseline models for information retrieval and summarization.

Rosa Stern & Alice Coucke: Music data processing for voice control

The focus of the Voice Experience team at Sonos is to bring together the profuse world of music and the slick user experience of a voice assistant within the Sonos home sound system. Supporting music related voice commands and a music catalog in our SLU (Spoken Language Understanding) system carries challenges at the various stages of our pipeline, which we'll present and discuss in this talk. We'll bring our focus on the main issues we encounter in our data processing pipeline, especially related to speech and voice recognition.

Isaac Julien & Shuo Zhang: Building a Personalized Voice Assistant for Music

The Bose Music Assistant was a former year-long research project that focused on building a personalized, conversational voice interface for music, with the goal of helping our customers find the content they enjoy. We will discuss the creation of a hybrid Grammar- and ML-based NLU engine that supported the Assistant and allowed us to quickly prototype and expand the experiences that it offered. We will also describe some of the NLP challenges we encountered in the music domain, and the opportunity that these challenges provided for personalization.

Table of Contents

Discovering Music Relations with Sequential Attention	1
<i>Junyan Jiang, Gus Xia and Taylor Berg-Kirkpatrick</i>	
Lyrics Information Processing: Analysis, Generation, and Applications	6
<i>Kento Watanabe and Masataka Goto</i>	
Did You ”the Next Episode? Using Textual Cues for Predicting Podcast Popularity	13
<i>Brihi Joshi, Shravika Mittal and Aditya Chetan</i>	
Using Latent Semantics of Playlist Titles and Descriptions to Enhance Music Recommendations	18
<i>Yun Hao and J. Stephen Downie</i>	
Prediction of user listening contexts for music playlists	23
<i>Jeong Choi, Anis Khlif and Elena Epure</i>	
An Information-based Model for Writing Style Analysis of Lyrics	28
<i>Melesio Crespo-Sanchez, Edwin Aldana-Bobadilla, Ivan Lopez-Arevalo and Alejandro Molina-Villegas</i>	
Generation of lyrics lines conditioned on music audio clips	33
<i>Olga Vechtomova, Gaurav Sahu and Dhruv Kumar</i>	
Hyperbolic Embeddings for Music Taxonomy	38
<i>Maria Astefanoaei and Nicolas Collignon</i>	
Computational Linguistics Metrics for the Evaluation of Two-Part Counterpoint Generated with Neural Machine Translation	43
<i>Stefano Kalonaris, Thomas McLachlan and Anna Aljanaki</i>	
Interacting with GPT-2 to Generate Controlled and Believable Musical Sequences in ABC Notation	49
<i>Cariña Geerlings and Albert Meroño-Peñuela</i>	
BUTTER: A Representation Learning Framework for Bi-directional Music-Sentence Retrieval and Generation	54
<i>Yixiao Zhang, Ziyu Wang, Dingsu Wang and Gus Xia</i>	
Unsupervised Melody Segmentation Based on a Nested Pitman-Yor Language Model	59
<i>Shun Sawada, Kazuyoshi Yoshii and Keiji Hirata</i>	
MusicBERT - learning multi-modal representations for music and text	64
<i>Federico Rossetto and Jeff Dalton</i>	
Music autotagging as captioning	67
<i>Tian Cai, Michael I Mandel and Di He</i>	
Comparing Lyrics Features for Genre Recognition	73
<i>Maximilian Mayerl, Michael Vötter, Manfred Moosleitner and Eva Zangerle</i>	
Classification of Nostalgic Music Through LDA Topic Modeling and Sentiment Analysis of YouTube Comments in Japanese Songs	78
<i>Kongmeng Liew, Yukiko Uchida, Nao Maeura and Eiji Aramaki</i>	

Discovering Music Relations with Sequential Attention

Junyan Jiang¹, Gus G. Xia¹, Taylor Berg-Kirkpatrick²

¹Music X Lab, NYU Shanghai

²Department of Computer Science and Engineering, University of California San Diego

{jj2731, gxia}@nyu.edu, tberg@eng.ucsd.edu

Abstract

The element-wise attention mechanism has been widely used in modern sequence models for text and music. The original attention mechanism focuses on token-level similarity to determine the attention weights. However, these models have difficulty capturing sequence-level relations in music, including repetition, retrograde, and sequences. In this paper, we introduce a new attention module called the sequential attention (SeqAttn), which calculates attention weights based on the similarity between pairs of sub-sequences rather than individual tokens. We show that the module is more powerful at capturing sequence-level music relations than the original design. The module shows potential in both music relation discovery and music generation.¹

1 Introduction

Music is one type of sequential data with distinctive structures. Various kinds of similarity occur among different phrases of a single music piece. Many music relations are based on sequence-level similarity. For example, a modulated sequence describes a music relation where two phrases' rhythm is the same, but the pitches are shifted.

A well-known method to capture relations in a sequence is the transformer model (Vaswani et al., 2017). Transformer-based models have had recent success in sequence generation and representation learning for both text (Radford et al., 2019; Devlin et al., 2018) and music (Huang et al., 2018; Dhariwal et al., 2020).

The core mechanism of the transformer is the element-wise attention layer. The attention module allows information exchange between any tokens in the sequences. However, it is not

¹Code and pre-trained models are available at <https://github.com/music-x-lab/SeqAttn>

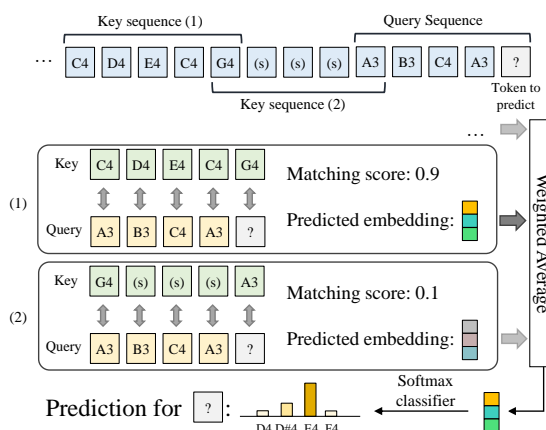


Figure 1: An overview of a self-attentive monophonic language model with the sequential attention mechanism. The model tries to predict the next token (represented by a question mark) by attending to related sub-sequences appear previously. (1) and (2) show two potential alignments. The model assigns a larger weight (matching score) to key sequence (1) over (2) since key sequence (1) has strong relations (tonal sequence) with the query sequence and can help to predict the next token (E4 in this case).

an explicit inductive bias for direct sequence-to-sequence matching. Second, a multi-layer attention setting is required: the model needs to collect the sequential information using the positional embedding (Vaswani et al., 2017; Shaw et al., 2018) on the first layer, and then compare the sequential information on the subsequent layers. These problems make the model hard to train and require additional parameters, which may also harm the model's generalization ability.

In this paper, we propose the sequential attention module, a new attention module that explicitly models sequence-level music relations. In this module, we measure the similarity of two sequences by a token-wise comparison instead of the dynamic time warping approach (Walder and

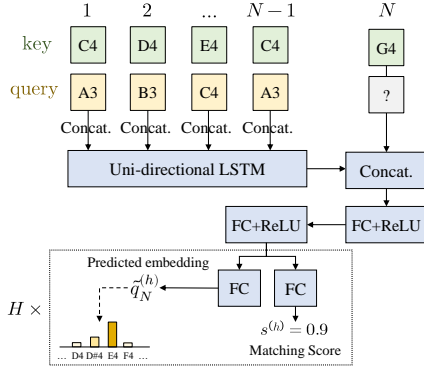


Figure 2: The architecture of the sequential attention unit with H heads. The model takes in the key and the query sequence and outputs the matching scores $s^{(h)}$ and the predicted embedding $\tilde{q}_N^{(h)}$ for each head.

Kim, 2018; Hu et al., 2003) to ensure time efficiency. We also show how to build a self-attentive language model based on the module to capture phrase-level self-similarity in a music piece. An overview of the process is shown in Figure 1. We show by experiments that the proposed model is better at capturing such self-similarity than the transformer model with a comparable size.

2 Proposed Method

2.1 Sequential Attention Unit

We first introduce the basic unit of the proposed module. The design is shown in Figure 2. Assume that we have two sequences of equal length, the query sequence $\mathbf{q} = (q_1, q_2, \dots, q_N)$ and the key sequence $\mathbf{k} = (k_1, k_2, \dots, k_N)$. Each q_n, k_n is an embedding vector of dimension d_v^s . Here, q_N is unknown while $\mathbf{k}_{1..N}$ and $\mathbf{q}_{1..N-1}$ are known. The target of the unit is to (1) estimate their matching score (s) between \mathbf{q} and \mathbf{k} , and (2) if they are well matched, predict the unknown element q_N given the corresponding key element k_N .

The module uses a multi-head setting (Vaswani et al., 2017) to allow learning multiple distinct relations between the same \mathbf{q}, \mathbf{k} pair. For a sequential attention unit with H attention heads, we have:

$$[s^{(1..H)}; \tilde{q}_N^{(1..H)}] = \text{SeqAttn}(\mathbf{q}_{1..N-1}, \mathbf{k}_{1..N}, e) \quad (1)$$

where e is a relative positional embedding vector. We first concatenate the corresponding elements in the query and key sequences, as well as the relative positional embedding ($f_n = [q_n; k_n; e]$), and feed them to a uni-directional LSTM. The last hidden state h_n and the last key element k_n are used to estimate the matching score $s^{(h)}$ and the predicted

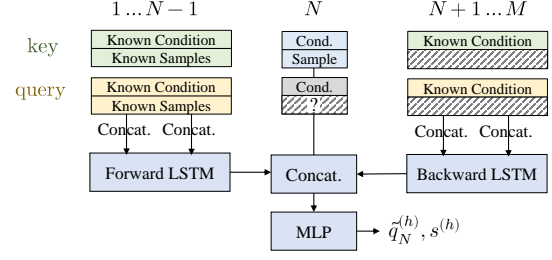


Figure 3: The architecture of the conditional sequential attention unit with H heads. The Multi-Layer Perceptron (MLP) has the same architecture as the unconditioned module.

$\tilde{q}_N^{(h)}$ for each head $h = 1..H$:

$$h_N = \text{LSTM}(f_1, f_2, \dots, f_{N-1}) \quad (2)$$

$$[s^{(1..H)}; \tilde{q}_N^{(1..H)}] = \text{MLP}([h_N; k_N]) \quad (3)$$

where MLP is a multi-layer perceptron with 3 fully connected layers and Rectified Linear Unit (ReLU) activations.

2.2 Self-Attention Layer

We now consider how to integrate the module into a language model using self-attention. Self-attention is a method to generate new tokens in a sequence by attending to previously generated ones. Given a partial sequence $x_{1..N-1}$, we want to predict x_N . We first enumerate the distance $i = 1, 2, 3, \dots$ between the query sequence and the key sequence. For each i , we calculate the matching score s_i and the predicted embedding $\tilde{x}_{N,i}$:

$$[s_i^{(1..H)}; \tilde{x}_{N,i}^{(1..H)}] = \text{SeqAttn}(\mathbf{x}_{1..N-1}, \mathbf{x}_{1-i..N-i}, e_i) \quad (4)$$

where e_i is a learned relative positional embedding for distance i . We will assign $x_k = \mathbf{0}$ for all non-positive indices $k \leq 0$. Then, a weighted average of $\tilde{x}_{N,i}$ is calculated as a final prediction. For each head $h = 1..H$, we have:

$$\hat{s}_i^{(h)} = \frac{\exp(s_i^{(h)})}{\sum_{i'} \exp(s_{i'}^{(h)})} \quad (5)$$

$$\tilde{x}_N^{(h)} = \sum_i \hat{s}_i^{(h)} \tilde{x}_{N,i}^{(h)} \quad (6)$$

$$\tilde{x}_N = \text{Linear}([\tilde{x}_N^{(1)}; \dots; \tilde{x}_N^{(H)}]) \quad (7)$$

We can then use $\text{Softmax}(\text{Linear}(\tilde{x}_N))$ to predict the probability of the actual tokens for x_n .

In practice, we do not enumerate all i values since most of the alignments do not agree with the rhythmic structure, thus less meaningful to perform the comparison. We can eliminate such cases to make the model more efficient. See section 3.2 for a detailed setting.

2.3 Conditional Sequential Attention

For the conditional sequence generation task, we propose the modified sequence attention unit, as shown in Figure 3. Here, we want to generate a target sequence x^s given a known condition sequence x^c (e.g., to generate the melody given the chord sequence). The major modification is that we add a backward LSTM to match the future conditions in order to generate the current token.

Assume we have the query sequence $\mathbf{q} = (q_1, q_2, \dots, q_M)$ and the key sequence $\mathbf{k} = (k_1, k_2, \dots, k_M)$ of equal length M . Each $q_n = (q_n^c, q_n^s)$ and $k_n = (k_n^c, k_n^s)$ are now a tuple of the sample and the condition. We assume that all conditions $k_{1..M}^c$, $q_{1..M}^c$ and a part of the samples $k_{1..N}^s$, $q_{1..N-1}^s$ are known ($N \leq M$). We are interested in estimating q_N^s . In this case, we change the Eqn. 2 and 3 to the following:

$$\vec{h}_N = \text{LSTM}_{\text{fw}}(f_1, f_2, \dots, f_{N-1}) \quad (8)$$

$$\overleftarrow{h}_N = \text{LSTM}_{\text{bw}}(b_M, b_{M-1}, \dots, b_{N+1}) \quad (9)$$

$$[s^{(1..H)}; \tilde{q}_N^{(1..H)}] = \text{MLP}([\vec{h}_N; \overleftarrow{h}_N; k_N; q_N^c]) \quad (10)$$

where $f_n = [k_n; q_n; e]$ and $b_n = [k_n^c; q_n^c; e]$. The forward LSTM tries to match the previous samples and the conditions, while the backward LSTM tries to match the future conditions only.

3 Experiments

3.1 Dataset

We trained and evaluated the proposed method on two datasets of different genres: (1) the Nottingham dataset (Foxley, 2011), an American folk dataset with 1,021 songs after filtering; (2) the POP dataset, a privately collected dataset with 1,394 Chinese pop songs with a 4/4 meter. All songs have a monophonic melody line with chord labels. For each dataset, we use 80% songs for training, 10% for validation, and 10% for testing. We augment the training set by pitch-shifting within the range [-5,6] semitones.

We quantize all songs to a sixteenth-note level. We represent each melody token as one of the 130 states: 128 onset states (for the 0-127 MIDI pitch range), 1 *sustain* state and 1 *silence* state. Each chord is encoded into a 36-dimensional multi-hot vector: 12 dimensions for the root scale, 12 dimensions for the bass scale, and 12 dimensions for its pitch classes.

Model	Nottingham		POP	
	Acc.	Ppl.	Acc.	Ppl.
Unconditioned models				
Mode	61.04	-	52.26	-
Ours+BA	88.23	1.54	84.08	1.77
Ours+MA	-	-	79.24	2.09
Transformer	84.58	1.70	70.69	2.73
Chord-conditioned models				
Ours+BA	90.26	1.40	85.27	1.68
Ours+MA	-	-	82.44	1.88
Transformer	84.87	1.66	71.30	2.61

Table 1: The comparative results for the accuracy and the perplexity of next token prediction on test sets.

3.2 Model Training

We implement both the conditional and unconditional models using sequential attention with $H = 4$ attention heads. We use $d_v^s = 256$ as the note embedding dimension, $d_v^c = 128$ as the chord embedding dimension, and $d_{\text{hidden}} = 256$ as the hidden dimension of the LSTM and MLP layers.

As mentioned in section 2.2, we only select the distance values i that leads to rhythmic meaningful alignments:

$$i \in \{i \in \mathbb{Z} | k \bmod i = 0 \text{ or } i \bmod k = 0\} \quad (11)$$

where k is a pre-defined group size. We experimented on two different selections: $k = 4$ for beat-level alignment (BA) and $k = 16$ for measure-level alignment (MA, only for 4/4 meter songs). For the Nottingham dataset, we only use beat-level alignment since it contains meter changes.

We define the model loss as the cross-entropy loss for the next token prediction task. The model is trained using the Adam optimizer (Kingma and Ba, 2014) with a constant learning rate of 1e-4. The training is stopped when the validation loss is not improved in 20 epochs.

To increase the robustness of the model performance, we randomly drop key-value pairs with a probability of 50% during training to encourage the model to discover more relations in a piece. The attention dropout is not used in testing.

3.3 Comparative Results

We first compare the proposed method against baseline methods for the next token prediction task. To predict the next token in a partial phrase, it is beneficial if the model learns to attend to sim-

	Input Sequence	Prediction	Ref.
(1)	A4 (s) B4 (s) C5 (s) G4 (s) F4 (s) (s) (s) E4 (s) (s) (s)	E4: 89.40%	E4
	A4 (s) B4 (s) C5 (s) G4 (s) F4 (s) (s) (s) ?	D4: 2.20%	
(2)	G4 (s) A4 (s) G4 (s) F4 (s) E4 (s) D4 (s) C4 (s) (s) (s)	Bb3: 51.24%	B3
	F4 (s) G4 (s) F4 (s) E4 (s) D4 (s) C4 (s) ?	B3: 36.85%	
(3)	C4 (s) D4 (s) E4 (s) F4 (s) G4 (s) E4 (s) C4 (s) G3 (s)	A3: 21.85%	A3
	D4 (s) E4 (s) F#4 (s) G4 (s) A4 (s) F#4 (s) D4 (s) ?	(s): 14.68%	

Table 2: A case study of the module’s behavior for different music relations: (1) exact repetition, (2) tonal sequence and (3) modulating sequence. The question mark is the token to predict and the (s) token is the *sustain* label. The table shows the top two predictions and their probability from the sequential attention model.

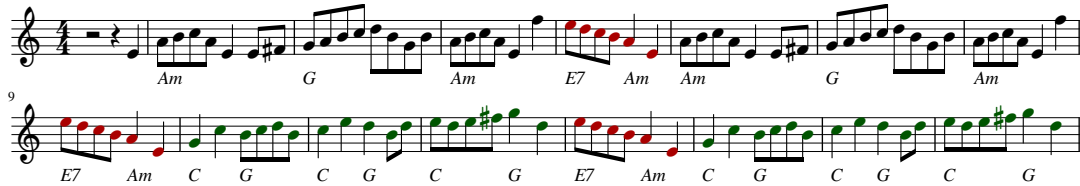


Figure 4: A generated sample. All chords and the melody for the first 8 bars are given. The model generates the melody for the next 8 bars. The repetitions in the generated piece are painted in colors (green and red).

ilar phrases appear previously. We use two baseline methods: (1) a weak baseline (Mode) that always predicts the most frequent token (the *sustain* label), and (2) a 3-layer transformer model with relative positional embedding (Shaw et al., 2018). The model has a transformer width of 256 and 4 attention heads. The results are listed in table 1. Results show that our proposed method acquires higher accuracy and lower perplexity on both the unconditioned model and the chord-conditioned model.

3.4 Analysis of the Attention Module

To further investigate the types of music relations that the sequential attention module captures, we apply the unconditional model with measure-level alignment to three 2-bar test cases with different music relations: (1) exact repetitions (2) tonal sequences and (3) modulating sequences, as shown in table 2. The model predicts reasonable results for all three test cases. Notice that the top 2 predictions of case (2) both form valid tonal sequences (in C major and F major keys, respectively). The model learns such music relations through self-supervision without explicit human instructions.

3.5 Music Generation

We also perform a music generation task using the conditioned language model. Figure 4 shows a generated example where we generate the next 8 bars of melody according to the chords and the first 8 bars of the melody of a sample (reelsd-

g18.mid) from the Nottingham test set. In this example, the model learns to repeat the phrases with the same chord sequences and to vary if the chord sequences changes.

However, as the model only performs token-by-token prediction, it lacks control over the global music structure. We found some generated examples have too many repetitions or too early cadences. Generating music with controlled music structures are left as a future work.

4 Conclusion

In this paper, we propose a new attention module, the sequential attention module, that explicitly models similarity between two sequences. Based on the module, we implement a self-attentive music language model. The model discovers and captures the self-similarity in music pieces and improves the next token prediction results.

Several important tasks are left as future works. First, the proposed method cannot capture music relations of different time scales since the sequential attention module performs a token-wise alignment of the query and the key sequence. A different module design is required in this case. Second, we want to explore whether the discovered relations can help us in other analysis and generation tasks, e.g., automatic music segmentation, and automatic music accompaniment.

References

- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. 2020. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*.
- E. Foxley. 2011. [Nottingham database](#).
- Ning Hu, Roger B Dannenberg, and George Tzanetakis. 2003. Polyphonic audio matching and alignment for music retrieval. In *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE Cat. No. 03TH8684)*, pages 185–188. IEEE.
- Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, Monica Dinulescu, and Douglas Eck. 2018. Music transformer. *arXiv preprint arXiv:1809.04281*.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.
- Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.
- Christian J Walder and Dongwoo Kim. 2018. Neural dynamic programming for musical self similarity. *arXiv preprint arXiv:1802.03144*.

Lyrics Information Processing: Analysis, Generation, and Applications

Kento Watanabe and Masataka Goto

National Institute of Advanced Industrial Science and Technology (AIST)

{kento.watanabe, m.goto}@aist.go.jp

Abstract

In this paper we propose *lyrics information processing (LIP)* as a research field for technologies focusing on lyrics text, which has both linguistic and musical characteristics. This field could bridge the natural language processing field and the music information retrieval field, leverage technologies developed in those fields, and bring challenges that encourage the development of new technologies. We introduce three main approaches in LIP, 1) lyrics analysis, 2) lyrics generation and writing support, and 3) lyrics-centered applications, and briefly discuss their importance, current approaches, and limitations.

1 Introduction

For songs that are musical pieces with singing voices, lyrics text is one of key factors that make listeners feel songs are attractive because it delivers messages and expresses emotion. Since the lyrics text plays an important role in music listening and creation, some studies in the music information retrieval (MIR) community have already focused on it, but not as many as studies that have focused on musical audio signals and musical scores. Similarly, in the natural language processing (NLP) community there have not been many studies focusing on lyrics text, and most NLP methods assume prose text, not lyrics text. Since lyrics text is a series of words, some NLP methods could be applied to it successfully, but NLP methods are not always effective for lyrics text because the natures of lyrics and prose texts are different as described in Section 2.

We therefore propose to refer to a broad range of lyrics-related studies as *lyrics information processing (LIP)*, which could also be considered music information processing for lyrics texts. LIP shares some core technologies with NLP and MIR, and research and development of LIP could contribute

to the MIR and NLP communities as follows:

(1) Academic contributions: Since lyrics are an important aspect of music information, LIP could broaden the scope of MIR and complement it. Since lyrics are a difficult form of natural language, LIP could provide challenging issues that are not addressed by existing NLP technologies. The nature of lyrics (e.g., style, structure, and semantics) could also be investigated by automatically analyzing and generating lyrics text data.

(2) Industrial contributions: LIP could open up practical applications that are useful for listeners and creators, such as lyrics classification, lyrics exploration, lyrics summarization, and lyrics writing support.

This paper gives an overview of LIP by categorizing lyrics-related studies into three main approaches: lyrics analysis, lyrics generation, and applications. Since the concept of LIP is broad and still emerging, we hope that this paper could stimulate further development of LIP.

2 Lyrics analysis

Because lyrics and poetry¹ have unique linguistic properties, NLP technologies for prose text are not always effective enough to analyze lyrics text. In this section we introduce studies of lyrics analysis regarding the structure and semantics of lyrics and its relationship with audio.

2.1 Lyrics structure analysis

Rhyme scheme identification: The rhyme scheme is the pattern of rhymes at the end of lyric lines. It is usually represented by using a series of letters corresponding to lines, in which repeated letters indicate rhymed lines. In the following example (RWC-MDB-P-2001 No.83 (Goto et al., 2002)),

¹Lyrics and poetry are different types of text because lyrics are assumed to be sung along with music. However, some linguistic properties of lyrics and poetry overlap.

two consecutive lines having the same letter rhyme:

A: *Race the clock I got to score*

A: *Work or play Back for more*

B: *Only true believers rise to the top,*

B: *Licking the cream of the crop*

This rhyme scheme is “AABB” and is called *Couplet*². Since typical prose text analyzers such as part-of-speech analyzers and grammar tree parsers cannot analyze rhyme schemes, some studies addressed the rhyme scheme identification task. Given a few lines of lyrics (paragraph or stanza) as the input, their rhyme scheme (ABC label sequence) is estimated. For example, Reddy and Knight (2011) and Addanki and Wu (2013) estimated the rhyme scheme by using language-independent unsupervised methods (e.g., hidden Markov models) that do not depend on morphological and phonological properties.

Lyrics segmentation: While the rhyme scheme is a line-by-line repetitive structure, lyrics also have a paragraph-by-paragraph structure like verse-bridge-chorus. Paragraphs are usually separated by a blank line, but in some lyrics they are not. Some studies therefore tackled the lyrics segmentation task in which the boundaries between paragraphs are estimated from lyrics without blank lines (Watanabe et al., 2016; Fell et al., 2018). They showed that the self-similarity matrix, which is often used in music structure analysis of audio signals in the MIR community, can be applied to lyrics text to improve the performance of lyrics segmentation. This is a good example of integrating NLP and MIR methods to accomplish a LIP task.

Verse-bridge-chorus labeling: Given paragraphs of lyrics, assigning a structural label such as verse, bridge, and chorus to each paragraph is also an important task. Simple rule-based methods such as a method of grouping paragraphs with the same label (Baratè et al., 2013) and a method of labeling each paragraph (Mahedero et al., 2005) have been proposed. Since a sufficient amount of lyrics data annotated with structural labels is still lacking for machine-learning approaches, there is much room for improvement.

2.2 Lyrics semantic analysis

Emotional expressions, topics, and stories in lyrics are factors that have a great influence on listeners’ emotions. Since lyrics tend to be constrained

²There are various rhyme schemes, such as ABAB (*Alternate Rhyme*), ABABBCBC (*Ballade*), AAAAA (*Monorhyme*), AAABBB (*Triplet*), and ABBA (*Enclosed Rhyme*).

by melody lines and have a limited length, a typical way of expressing messages in lyrics is different from the way they are expressed in prose text. Lyrics messages are often emotional, inspiring, concise, and (intentionally) obscure. Even if detailed moods, topics, and stories are not explicitly described in lyrics, listeners can enjoy guessing or inferring them. Some studies have already analyzed such semantic factors behind lyrics text.

Mood estimation: Supervised learning-based methods estimating the mood or emotion of lyrics have been developed (Wang et al., 2011; Hu and Downie, 2010; Delbouys et al., 2018) and are based on a word dictionary in which valence and arousal values (Russell, 2003) are annotated (Bradley and Lang, 1999; Warriner et al., 2013). Since a lot of mood estimation methods for audio signals have been proposed in the MIR community, it would be possible to develop mood estimation based on both lyrics text and audio. In the future, unsupervised methods and support for low-resource languages are expected to be developed because supervised learning-based methods require training data of annotated lyrics, which are language-dependent.

Topic modeling: For lyrics topic modeling, unsupervised methods such as latent Dirichlet allocation (LDA), non-negative matrix factorization, and their extensions are often used (Kleedorfer et al., 2008; Sasaki et al., 2014; Tsukuda et al., 2017). Unlike mood estimation methods, these methods do not require training data with valence and arousal values, which results in the advantage of easily preparing training data for different languages. The obtained word topics (clusters) are further used as clues for classification tasks or used in visualization functions for music exploration. It is, however, difficult to appropriately evaluate the accuracy of topics obtained by unsupervised learning. A previous study tackled this difficulty by evaluating the correlation between estimated topics clusters and human-annotated ones (Sterckx et al., 2014).

Storyline modeling: Lyric writers consider themes and stories when writing lyrics. For the verse-bridge-chorus structure of lyrics, an example of a storyline represented as a topic transition is *introduction* (verse) → *past event* (bridge) → *emotional message* (chorus). Watanabe et al. (2018b) proposed an extended hidden Markov model to learn this topic transition structure from lyrics data without supervision. Their model learned topic transitions that are often found in love songs, hip-

hop songs, and so on even if they are not explicitly given.

2.3 Analysis of the relationship between lyrics text and music audio

A clear difference between lyrics and poetry is the presence or absence of accompanying music. Since investigating the relationship and synchronization between lyrics and music audio is an important topic of research, there have been various related studies that deal with the relationship between syllable stress and pitch (Nichols et al., 2009), the relationship between words and chords (Greer et al., 2019), the relationship between rests in melody and boundaries of words, lines, and paragraphs (Watanabe et al., 2018a), and lyrics-to-audio alignment (Kan et al., 2008; Fujihara et al., 2011; Mauch et al., 2012; Chien et al., 2016; Chang and Lee, 2017; Stoller et al., 2019; Gupta et al., 2019).

3 Lyrics generation and writing support

As natural language generation (NLG) has been actively researched, automatic lyrics generation is becoming a popular topic of research. NLG technologies have been greatly improved in performance by deep neural networks (DNNs) and are utilized in applications such as machine translation and dialogue systems. Generating poetry and novels has also been developed, though generating creative text is challenging. Generating lyrics is also challenging and has further technical difficulties caused by lyrics-specific musical constraints such as melodies and rhymes. In this section we introduce studies of lyrics generation as well as writing support systems that utilize lyrics generation methods.

3.1 Automatic lyrics generation

Rhyme-scheme-conditioned lyrics generation: Since lyrics and poetry often have rhyme schemes as introduced in Section 2.1, some studies have addressed the task of generating lyrics and poetry that satisfy constraints of a rhyme scheme (Barbieri et al., 2012; Hopkins and Kiela, 2017). In automatically generating lyrics, most methods use language models such as n-grams and recurrent neural networks as well as word sequence search based on the Markov process. To deal with the constraints, several extended word-sequence search methods have been proposed, such as those using the strong constraint that words that do not satisfy the rhyme

scheme are discarded during word sequence search and the weak constraint that the score is calculated based on how well the given rhyme scheme is satisfied.

Melody-conditioned lyrics generation: Although most studies of automatic lyrics generation have generated lyrics using only text data without considering musical audio signals and musical scores, some studies have addressed the task of generating fluent lyrics that are singable when a melody (a sequence of musical notes) is given (Lu et al., 2019). Watanabe et al. (2018a) confirmed that the frequency of word/line/paragraph boundaries depends on the duration of rests and proposed an advanced lyrics language model that takes advantage of this dependency. Their method can generate segmented lyrics that are singable for the verse-bridge-chorus structure of the input melody. It, however, requires training data in which lyrics syllables and melody notes are aligned. Such data could be easily created if technologies such as the above-mentioned lyrics-to-audio alignment, lyrics recognition (Hosoya et al., 2005; Dabike and Barker, 2019; Suzuki et al., 2019), and melody note transcription (Yang et al., 2017; Román et al., 2018; Nishikimi et al., 2019) could mature in the future.

Automatic generation of structured lyrics: Most lyrics generation systems can generate only one paragraph of lyrics, though lyrics have some paragraphs in general. This is because language models for lyrics did not explicitly capture the consistency of topics and relations between paragraphs. Watanabe et al. (2014) have proposed a probabilistic model that captures topic transitions between paragraphs to generate lyrics having the storyline. Fan et al. (2019) have proposed a lyrics generation method using the long short-term memory language model that captures the hierarchical structure of words, lines, and paragraphs to leverage the dependency of long word sequences. Although these studies have made it possible to generate lyrics that are almost consistent in topic, it is still difficult to generate lyrics that are consistent in meaning.

Ghostwriting: Ghostwriting is a task of generating new lyrics that follow the style (e.g., rhyme scheme, phrasing, content, and the number of words per line) of a given artist. Potash et al. (2015) proposed a rap-lyrics generation method based on data-driven learning of the artist’s style using a DNN-based language model trained with the artist’s lyrics corpus.

3.2 Writing support system with automatic lyrics generation

Automatic lyrics generation makes it possible to develop systems that support lyrics writing. It is not easy for novices to write lyrics by thinking of appropriate words and phrases while considering various constraints and properties. Since candidate word sequences satisfying various constraints can be generated automatically, it is useful to show them to lyric writers to support their creative activities. Some studies have developed interactive systems that support lyrics writing by repeatedly recommending candidate word sequences that satisfy constraint parameters input by the user.

pâtissier (Abe and Ito, 2012) is an interface that allows the user to specify syllable counts, syllable stress, and vowels, and generates candidate sentences that satisfy them. *DeepBeat* (Malmi et al., 2016) is an interface that generates and suggests next-line candidates that rhyme with a line entered by the user. *LyriSys* (Watanabe et al., 2017) and *Co-PoeTryMe* (Oliveira et al., 2019) are interfaces that allow the user to specify song structure and syllable counts, select or enter topics and keywords for each paragraph, and make the system generate candidate lyrics that satisfy them. These interfaces also allow the user to manually edit the generated lyrics.

4 Applications for a collection of lyrics

Like NLP technologies, LIP technologies are useful in developing various applications, such as classification, exploration, and summarization, for a large collection of lyrics data.

4.1 Lyrics classification

Given a collection of lyrics, it is useful to classify and visualize them. Genre classification for lyrics is a popular approach that has already been studied (Mayer et al., 2008; Mayer and Rauber, 2011; Tsaptsinos, 2017). Some characteristics peculiar to lyrics (e.g., rhyme scheme, structure, meaning, and relationship with audio) have been used as features to train a supervised classifier.

4.2 Lyrics exploration

If a user wants to see the lyrics of a song the user knows, simple text-based lyrics retrieval is enough, but if a user wants to encounter unfamiliar but interesting lyrics, a content-based music exploration system focusing on lyrics is necessary. Baur et al.

(2010), Sasaki et al. (2014), and Tsukuda et al. (2017) have developed such exploration systems that visualize topics of lyrics and similar artists by analyzing the content of lyrics using LDA, self-organizing maps, and so on. *Query-by-Blending* (Watanabe and Goto, 2019) is a music exploration system that enables a user to give flexible queries related to lyrics, audio signals, and artist tags by using a unified latent vector space with these three different modalities embedded.

4.3 Lyrics summarization

In browsing a collection of lyrics, a short summary of lyrics of each song helps navigate quickly. Fell et al. (2019) improved the performance of the lyrics summarization task by combining a general document summarization method with an audio thumbnailing method. Summarization more advanced than simply extracting lines, such as phrase paraphrasing and compression, requires development of advanced technologies for lyrics semantic analysis.

5 Conclusion

In this paper we have provided an overview of lyrics information processing (LIP) and have described examples of studies from the viewpoint of lyrics analysis, lyrics generation, and applications. Those examples are just excerpts taken from a variety of previous studies and possible future technologies. For example, the limited space does not allow us to discuss the relationship with *singing information processing (SIP)* (Goto et al., 2010; Goto, 2014; Humphrey et al., 2019), though we mentioned the lyric-to-audio alignment. Since lyrics are sung by singers, there are many possibilities to investigate the relationship between lyrics and the corresponding singing expressions and styles. Lyrics are thus linguistic, musical, and singable from the NLP, MIR, and SIP viewpoints, respectively. Since LIP is an emerging interdisciplinary research field that could be related to various technologies and disciplines such as natural language processing, music information retrieval, machine learning, human-computer interaction, visualization, signal processing, linguistics, and musicology, we expect research on LIP to progress in coming years from a diverse viewpoint by attracting more attention due to its importance and potential.

6 Acknowledgments

This work was supported in part by JST ACCEL Grant Number JPMJAC1602, and JSPS KAKENHI Grant Number 20K19878 Japan.

References

- Chihiro Abe and Akinori Ito. 2012. A Japanese lyrics writing support system for amateur songwriters. In *Proceedings of the Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC 2012)*, pages 1–4.
- Kartek Addanki and Dekai Wu. 2013. Unsupervised rhyme scheme identification in hip hop lyrics using hidden markov models. In *Proceedings of The First International Conference of the Statistical Language and Speech Processing (SLSP 2013)*, volume 7978, pages 39–50.
- Adriano Baratè, Luca A. Ludovico, and Enrica Santucci. 2013. A semantics-driven approach to lyrics segmentation. In *Proceedings of the 8th International Workshop on Semantic and Social Media Adaptation and Personalization (SMAP 2013)*, pages 73–79.
- Gabriele Barbieri, François Pachet, Pierre Roy, and Mirko Degli Esposti. 2012. Markov constraints for generating lyrics with style. In *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI 2012)*, volume 242, pages 115–120.
- Dominikus Baur, Bartholomäus Steinmayr, and Andreas Butz. 2010. SongWords: Exploring music collections through lyrics. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, pages 531–536.
- Margaret M. Bradley and Peter J. Lang. 1999. Affective norms for English words (ANEW): Instruction manual and affective ratings. Technical report, Technical report C-1, the center for research in psychophysiology.
- Sungkyun Chang and Kyogu Lee. 2017. Lyrics-to-audio alignment by unsupervised discovery of repetitive patterns in vowel acoustics. *IEEE Access*, 5:16635–16648.
- Yu-Ren Chien, Hsin-Min Wang, and Shyh-Kang Jeng. 2016. Alignment of lyrics with accompanied singing audio based on acoustic-phonetic vowel likelihood modeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(11):1998–2008.
- Gerardo Roa Dabike and Jon Barker. 2019. Automatic lyric transcription from karaoke vocal tracks: Resources and a baseline system. In *Proceedings of the 20th Annual Conference of the International Speech Communication Association (Interspeech 2019)*, pages 579–583.
- Rémi Delbouys, Romain Hennequin, Francesco Piccoli, Jimena Royo-Letelier, and Manuel Moussallam. 2018. Music mood detection based on audio and lyrics with deep neural net. In *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR 2018)*, pages 370–375.
- Haoshen Fan, Jie Wang, Bojin Zhuang, Shaojun Wang, and Jing Xiao. 2019. A hierarchical attention based seq2seq model for Chinese lyrics generation. In *Proceedings of the 16th Pacific Rim International Conference on Artificial Intelligence (PRICAI 2019)*, pages 279–288.
- Michael Fell, Elena Cabrio, Fabien Gandon, and Alain Giboin. 2019. Song lyrics summarization inspired by audio thumbnailing. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 328–337.
- Michael Fell, Yaroslav Nechaev, Elena Cabrio, and Fabien Gandon. 2018. Lyrics segmentation: Textual macrostructure detection using convolutions. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018)*, pages 2044–2054.
- Hiromasa Fujihara, Masataka Goto, Jun Ogata, and Hiroshi G. Okuno. 2011. LyricSynchronizer: Automatic synchronization system between musical audio signals and lyrics. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1252–1261.
- Masataka Goto. 2014. Singing information processing. In *Proceedings of the 12th IEEE International Conference on Signal Processing (IEEE ICSP 2014)*, pages 2431–2438.
- Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. 2002. RWC Music Database: Popular, classical, and jazz music databases. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR 2002)*, pages 287–288.
- Masataka Goto, Takeshi Saitou, Tomoyasu Nakano, and Hiromasa Fujihara. 2010. Singing information processing based on singing voice modeling. In *Proceedings of the 2010 IEEE International Conference on Acoustics, Speech, and Signal Processing (IEEE ICASSP 2010)*, pages 5506–5509.
- Timothy Greer, Karan Singla, Benjamin Ma, and Shrikanth S. Narayanan. 2019. Learning shared vector representations of lyrics and chords in music. In *Proceedings of the 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (IEEE ICASSP 2019)*, pages 3951–3955.
- Chitralakha Gupta, Emre Yilmaz, and Haizhou Li. 2019. Acoustic modeling for automatic lyrics-to-audio alignment. In *Proceedings of the 20th Annual Conference of the International Speech Communication Association (Interspeech 2019)*, pages 2040–2044.

- Jack Hopkins and Douwe Kiela. 2017. Automatically generating rhythmic verse with neural networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 168–178.
- Toru Hosoya, Motoyuki Suzuki, Akinori Ito, and Shozo Makino. 2005. Lyrics recognition from a singing voice based on finite state automaton for music information retrieval. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR 2005)*, pages 532–535.
- Xiao Hu and J. Stephen Downie. 2010. When lyrics outperform audio for music mood classification: A feature analysis. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, pages 619–624.
- E. J. Humphrey, S. Reddy, P. Seetharaman, A. Kumar, R. M. Bittner, A. Demetriou, S. Gulati, A. Jansson, T. Jehan, B. Lehner, A. Krupse, and L. Yang. 2019. An introduction to signal processing for singing-voice analysis: High notes in the effort to automate the understanding of vocals in music. *IEEE Signal Processing Magazine*, 36(1):82–94.
- Min-Yen Kan, Ye Wang, Denny Iskandar, Tin Lay Nwe, and Arun Shenoy. 2008. LyricAlly: Automatic synchronization of textual lyrics to acoustic music signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):338–349.
- Florian Kleedorfer, Peter Knees, and Tim Pohle. 2008. Oh Oh Oh Whoah! towards automatic topic detection in song lyrics. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR 2008)*, pages 287–292.
- Xu Lu, Jie Wang, Bojin Zhuang, Shaojun Wang, and Jing Xiao. 2019. A syllable-structured, contextually-based conditionally generation of Chinese lyrics. In *Proceedings of the 16th Pacific Rim International Conference on Artificial Intelligence (PRICAI 2019)*, pages 257–265.
- Jose P. G. Mahedero, Alvaro Martinez, Pedro Cano, Markus Koppenberger, and Fabien Gouyon. 2005. Natural language processing of lyrics. In *Proceedings of the 13th ACM International Conference on Multimedia (ACM Multimedia 2005)*, pages 475–478.
- Eric Malmi, Pyy Takala, Hannu Toivonen, Tapani Raiko, and Aristides Gionis. 2016. DopeLearning: A computational approach to rap lyrics generation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 195–204.
- Matthias Mauch, Hiromasa Fujihara, and Masataka Goto. 2012. Integrating additional chord information into HMM-based lyrics-to-audio alignment. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):200–210.
- Rudolf Mayer, Robert Neumayer, and Andreas Rauber. 2008. Rhyme and style features for musical genre classification by song lyrics. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR 2008)*, pages 337–342.
- Rudolf Mayer and Andreas Rauber. 2011. Music genre classification by ensembles of audio and lyrics features. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, pages 675–680.
- Eric Nichols, Dan Morris, Sumit Basu, and Christopher Raphael. 2009. Relationships between lyrics and melody in popular music. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, pages 471–476.
- Ryo Nishikimi, Eita Nakamura, Masataka Goto, and Kazuyoshi Yoshii. 2019. End-to-end melody note transcription based on a beat-synchronous attention mechanism. In *Proceedings of the 2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE WASPAA 2019)*, pages 26–30.
- Hugo Gonçalo Oliveira, Tiago Mendes, Ana Boavida, Ai Nakamura, and Margareta Ackerman. 2019. CoPoeTryMe: Interactive poetry generation. *Cognitive Systems Research*, 54:199–216.
- 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 (EMNLP 2015)*, pages 1919–1924.
- Sravana Reddy and Kevin Knight. 2011. Unsupervised discovery of rhyme schemes. In *Proceedings of The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011)*, pages 77–82.
- Miguel A. Román, Antonio Pertusa, and Jorge Calvo-Zaragoza. 2018. An end-to-end framework for audio-to-score music transcription on monophonic excerpts. In *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR 2018)*, pages 34–41.
- James A. Russell. 2003. Core affect and the psychological construction of emotion. *Psychological review*, 110(1):145.
- Shoto Sasaki, Kazuyoshi Yoshii, Tomoyasu Nakano, Masataka Goto, and Shigeo Morishima. 2014. LyricsRadar: A lyrics retrieval system based on latent topics of lyrics. In *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR 2014)*, pages 585–590.
- Lucas Sterckx, Thomas Demeester, Johannes Deleu, Laurent Mertens, and Chris Develder. 2014. Assessing quality of unsupervised topics in song lyrics. In *Proceedings of the 36th European Conference on IR Research (ECIR 2014)*, volume 8416, pages 547–552.

- Daniel Stoller, Simon Durand, and Sebastian Ewert. 2019. End-to-end lyrics alignment for polyphonic music using an audio-to-character recognition model. In *Proceedings of the 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (IEEE ICASSP 2019)*, pages 181–185.
- Motoyuki Suzuki, Sho Tomita, and Tomoki Morita. 2019. Lyrics recognition from singing voice focused on correspondence between voice and notes. In *Proceedings of the 20th Annual Conference of the International Speech Communication Association (Interspeech 2019)*, pages 3238–3241.
- Alexandros Tsaptsinos. 2017. Lyrics-based music genre classification using a hierarchical attention network. In *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR 2017)*, pages 694–701.
- Kosetsu Tsukuda, Keisuke Ishida, and Masataka Goto. 2017. Lyric Jumper: A lyrics-based music exploratory web service by modeling lyrics generative process. In *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR 2017)*, pages 544–551.
- Xing Wang, Xiaoou Chen, Deshun Yang, and Yuqian Wu. 2011. Music emotion classification of Chinese songs based on lyrics using TF*IDF and rhyme. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, pages 765–770.
- Amy Beth Warriner, Victor Kuperman, and Marc Brysbaert. 2013. Norms of valence, arousal, and dominance for 13,915 English lemmas. *Behavior research methods*, 45(4):1191–1207.
- Kento Watanabe and Masataka Goto. 2019. Query-by-Blending: A music exploration system blending latent vector representations of lyric word, song audio, and artist. In *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR 2019)*, pages 144–151.
- Kento Watanabe, Yuichiroh Matsubayashi, Satoru Fukayama, Masataka Goto, Kentaro Inui, and Tomoyasu Nakano. 2018a. 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 (NAACL HLT 2018)*, pages 163–172.
- Kento Watanabe, Yuichiroh Matsubayashi, Kentaro Inui, Satoru Fukayama, Tomoyasu Nakano, and Masataka Goto. 2018b. Modeling storylines in lyrics. *IEICE Transactions on Information and Systems*, E101-D(4):1167–1179.
- Kento Watanabe, Yuichiroh Matsubayashi, Kentaro Inui, and Masataka Goto. 2014. Modeling structural topic transitions for automatic lyrics generation. In *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computation (PACLIC 2014)*, pages 422–431.
- Kento Watanabe, Yuichiroh Matsubayashi, Kentaro Inui, Tomoyasu Nakano, Satoru Fukayama, and Masataka Goto. 2017. LyriSys: An interactive support system for writing lyrics based on topic transition. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces (ACM IUI 2017)*, pages 559–563.
- Kento Watanabe, Yuichiroh Matsubayashi, Naho Orita, Naoaki Okazaki, Kentaro Inui, Satoru Fukayama, Tomoyasu Nakano, Jordan B. L. Smith, and Masataka Goto. 2016. Modeling discourse segments in lyrics using repeated patterns. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING 2016)*, pages 1959–1969.
- Luwei Yang, Akira Maezawa, Jordan B. L. Smith, and Elaine Chew. 2017. Probabilistic transcription of sung melody using a pitch dynamic model. In *Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (IEEE ICASSP 2017)*, pages 301–305.

Did You “Read” the Next Episode? Using Textual Cues for Predicting Podcast Popularity

Brihi Joshi* Shravika Mittal* Aditya Chetan*

Indraprastha Institute of Information Technology, Delhi

{brihi16142, shravika16093, aditya16217}@iiitd.ac.in

Abstract

Podcasts are an easily accessible medium of entertainment and information, often covering content from a variety of domains. However, only a few of them garner enough attention to be deemed ‘popular’. In this work, we investigate the textual cues that assist in differing popular podcasts from unpopular ones. Despite having very similar polarity and subjectivity, the lexical cues contained in the podcasts are significantly different. Thus, we employ a triplet-based training method, to learn a text-based representation of a podcast, which is then used for a downstream task of “popularity prediction”. Our best model received an F1 score of 0.82, achieving a relative improvement over the best baseline by 12.3%.

1 Introduction

Predicting the popularity of media content, such as songs, podcasts, etc., before its release can have significant implications for the producers, artists, etc. Traditionally, this task has been attempted with hand-crafted feature sets (Tsagkias et al., 2008), and utilising various audio features (Dhanaraj and Logan, 2005). However, hand-crafted feature sets are often not scalable, while audio-based features ignore the textual cues that are present in the data. Recently, with the rise in popularity and efficacy of Deep Learning, Neural network-based models (Yang et al., 2017; Zangerle et al., 2019) have also been proposed for hit-song prediction. There have also been some attempts (Yang et al., 2019) to learn a general representation for media content, but only based on the audio of the content, not from the textual cues.

In this work, we attempt to study the following: *How does the textual content of popular podcasts differ from that of unpopular ones?* First, we conduct experiments to assess the polarity of

popular podcasts, and observe that it is quite similar to that of unpopular podcasts. This observation is also prevalent while studying the subjectivity of the transcripts. Furthermore, there is little to no variation when polarity and subjectivity are studied over time. We then analyse the differences in the keywords and the general topical categories interspersed between popular and unpopular podcasts. It is observed that content generally centered around ‘Politics’, ‘Crime’ or ‘Media’ is more popular than others. Keeping this in mind, we design a triplet-training method, that leverages similarities between the popular and unpopular podcast samples to create representations that are useful in the downstream podcast popularity prediction task.

2 Related Work

The problem of “popularity prediction” has been explored for different types of media content, in a variety of ways. For instance, Hit song prediction has been an active area of research. Dhanaraj and Logan (2005) used spectral features like MFCCs to train an SVM for predicting whether a song would be a hit or not. Yang et al. (2017) proposed a Convolutional Neural Network based architecture for predicting the popularity of a song, using audio-based features. More recently, Zangerle et al. (2019) employed a combination of low-level and high-level audio descriptors for training Neural Networks on a regression task. However, these works have not taken textual cues into account when predicting the popularity of a song. Sanghi and Brown (2014) made an attempt to use lyric-based features that incorporated the rhyming quality of the song. However, they did not learn a representation based on the lyrics.

For podcasts, Tsagkias et al. (2008) gave a framework for assessing the credibility of pod-

*Equal contribution. Ordered randomly.

casts. Their notion of credibility included preference of the listeners. The framework was also shown to be reasonably effective in predicting popular podcasts (Tsagkias et al., 2009). This framework included highly refined hand-crafted features, based on both audio, textual and content describing the podcast on its platform. Recently, Yang et al. (2019) proposed a GAN-based model, for learning representations of podcasts, based on non-textual features, and showed its applications in downstream tasks like music retrieval and popularity prediction.

Finally, popularity prediction is also challenging because of the class imbalance that is inherent in the problem definition itself. Popular podcasts or songs would always be in a minority in a corpus. This makes the task of learning a good representation for them difficult. To overcome this, we exploit the triplet-based training procedure (Hoffer and Ailon, 2015) for generating a balanced distribution of both popular and unpopular podcasts as the “anchor” podcast. (See Section 5.1)

3 Dataset

In our study, we use the dataset collected by Yang et al. (2019) as a part of their podcast popularity prediction task. The dataset consists of 6511 episodes among which, there are 837 popular and 5674 unpopular (*long-tail*) podcasts. Based on the iTunes chart ranking, channels corresponding to the top 200 podcasts were treated as “top channels” and episodes from these top channels were then labelled as popular. Yang et al. (2019) provide a random 60-40 split of the dataset as a training and testing set. The average duration of the podcasts is 9.83 minutes.

In this work, we only use the transcripts that are provided with the podcast audio. Each transcript contains the start and end timestamps (in milliseconds) along with every spoken token in a new line. We remove the timestamps and stop words for all transcripts. We also do not consider non-verbal vocalisations in the transcript (for example, “oooo”, “ahhh”, etc.) for our analysis. After pre-processing, the podcast transcriptions contain 1557 tokens on an average.

4 Data Analysis

4.1 Polarity Analysis

In order to understand the general polarity and sentiment across popular and unpopular podcasts,

we extract the polarity scores of each podcast using TEXTBLOB¹, which is calculated by averaging the polarity of pre-defined lexicons, inferred from the words in the podcast. The polarity values range between -1 to 1 , where anything above 0 is considered to be ‘positive’.

We average the obtained polarity scores for all the podcasts, for each of the popular and unpopular categories. It was observed that the overall polarity of popular and unpopular podcasts is roughly **the same** – as the average polarity score for the popular class was 0.14 and for the unpopular class was 0.15 .

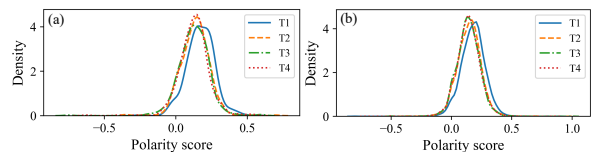


Figure 1: Density distribution of raw polarity scores for (a) Popular and (b) Unpopular podcasts over four time intervals.

In order to understand how polarity varies *over time*, we split each podcast into four time-chunks based on the three quartiles ($Q1$, $Q2$ and $Q3$), which we call $T1$, $T2$, $T3$ and $T4$, in order, with the help of the timestamps provided with the podcast transcripts.

Figure 1 shows the density distributions for raw polarity scores over the four splits (based on timestamps) for the two categories. It is observed that **both** popular and unpopular podcasts start-off with a positive tone, slowly transitioning into neutral content. However, there is limited observable distinction between popular and unpopular podcasts based on polarity.

4.2 Subjectivity Analysis

Similar to Polarity analysis, we looked into subjectivity scores for each podcast using TEXTBLOB, which is calculated by averaging the subjectivity of pre-defined lexicons, inferred from the words in the podcast. The values vary between 0 and 1 such that, the higher the score the more ‘opinion based’ (subjective) the text is.

As was observed for polarity, the overall subjectivity of popular and unpopular podcasts is exactly **the same** – as the average subjectivity score obtained across all podcasts was 0.48 for both popular and unpopular classes.

¹<https://textblob.readthedocs.io/en/dev/>

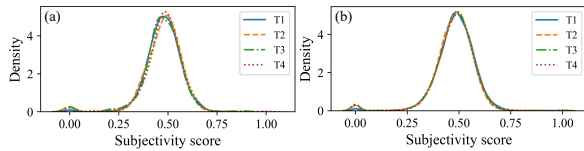


Figure 2: Density distribution of raw subjectivity scores for (a) Popular and (b) Unpopular podcasts over four time intervals.

To capture how subjectivity varies over time we used the same four timestamp based podcast chunks as was used for Polarity analysis. Figure 2 shows the density distributions for raw subjectivity scores over the four splits for the two categories.

It can again be observed that **both** popular and unpopular podcasts maintain their subjectivity over time with no significant differences across categories.

4.3 Lexical Analysis

We use EMPATH (Fast et al., 2016) to analyse the topical signals with the help of 194 pre-defined lexicons (for example – ‘social media’, ‘war’, ‘violence’, ‘money’, ‘alcohol’, ‘crime’ to name a few) that highly correlate with LIWC (Tausczik and Pennebaker, 2010).

We extract the scores from EMPATH for each category, for each podcast. The most and the least relevant lexical categories for popular podcasts, ordered by their significance values are given in Table 1.

Rank	Lexical Categories
1	Government
2	Crime
3	Politics
4	Money
5	Law
190	Hygiene
191	Social Media
192	Urban
193	Worship
194	Swimming

Table 1: Lexical Categories that are more likely to be present in popular podcasts, than unpopular podcasts: We run a Welch’s two sample t-test on the category scores for each podcast. Top-5 lexical categories shown are more significantly ($p < 0.05$) present in popular podcasts, than unpopular ones. Bottom-5 categories are ordered according to least significance ($p > 0.95$).

4.4 Keyword co-occurrence

We also study what kind of keywords are present in popular and unpopular podcasts. We rank bi-grams based on their Pointwise Mutual Information (PMI) scores and report the top 10 in Table 2.

It can be observed that in podcasts belonging to the popular class, keyword pairs like ‘Hillary Clinton’, ‘Donald Trump’, or ‘Gordon Hayward’ outshine highlighting the possibility of domain areas such as ‘Politics’, ‘Sports’, or ‘Celebrities’ to be responsible for making a podcast popular. This can also be seen in Section 4.3, which shows that ‘Government’ related topics are widely present in popular podcasts.

On the other hand the top keyword pairs extracted from unpopular podcasts belong to more generic domains like ‘Cities’, ‘Lifestyle’, etc., to name a few.

Popular		Unpopular	
Bi-gram	PMI	Bi-gram	PMI
Los Angeles	42.26	Web Site	85.62
United States	37.95	New York	80.63
New York	26.73	E Mail	80.50
Gordon Hayward	15.11	Fourth July	61.28
North Korea	14.56	Two Thousand	60.50
Blue Apron	13.87	High School	52.82
Hillary Clinton	12.40	Las Vegas	43.00
Donald Trump	9.02	Hong Kong	41.90
Fourth July	8.19	Real Estate	37.44
San Francisco	8.01	Wal Mart	34.71

Table 2: Top 10 bi-grams (ranked by their PMI values) for Popular vs. Unpopular podcasts: The keyword bi-grams in bold are encompassed by topics that are shown to be highly relevant for popular podcasts in Section 4.3.

5 Podcast Popularity Prediction

5.1 Proposed Method

Owing to the lack of a balanced dataset for popularity prediction, we use the Triplet Training strategy. In this method, instead of having class labels like ‘popular’ or ‘unpopular’ for the podcasts, we group the podcasts into triplets – each triplet has an anchor a podcast, which is often the reference for comparison, a positive podcast p which belongs to the same class as a , and a negative podcast n which belongs to the other class. The intuition is to reduce the distance between the representations of podcasts belonging to the same class and vice versa. After extracting the representation

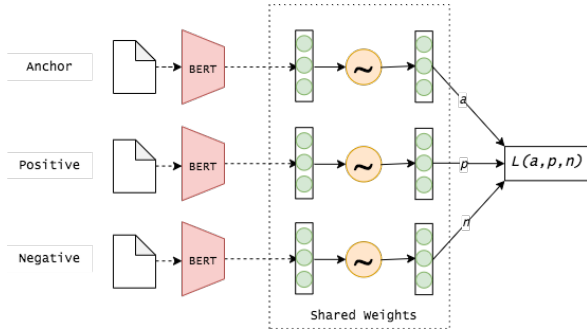


Figure 3: Triplet Training Architecture: The podcast triples are first passed through a DISTILBERT model, followed by a 2-layer Neural Network, with a RELU non-linearity in between. The weights are shared across the triplet during training.

of all the three podcasts in a triplet from a network with shared weights, we use the Triplet loss given below, as introduced by [Schroff et al. \(2015\)](#).

$$\mathcal{L}(a, p, n) = \sum_{i=1}^N [\|f(a_i) - f(p_i)\|_2^2 - \|f(a_i) - f(n_i)\|_2^2 + \alpha]$$

where a_i , p_i and n_i are the anchor, positive and negative podcast samples in the i^{th} triplet, f is a function that outputs an embedding for the podcasts and α is the margin between the positive and negative podcast samples.

We use a pre-trained DISTILBERT ([Sanh et al., 2019](#)) model² to create initial representations for the podcasts, followed by two fully connected layers, which shared weights during the triplet training phase. The architecture can be seen in Figure 3. The output of the final layer is a 128-dimensional vector, that is used as an embedding for the downstream popularity prediction task.

5.2 Evaluation and Results

The following methods are used to extract the representations of podcasts to predict their popularity:

- **TF-IDF:** TF-IDF weights ([Ramos et al., 2003](#)) corresponding to each word in a podcast are used to fill a vector, the size of which equals the size of training set’s vocabulary.
- **WORD2VEC (WV):** WORD2VEC ([Mikolov et al., 2013](#)) embeddings for each word in a podcast are averaged to create a single embedding representing the podcast.

²We use the DISTILBERT BASE model provided by huggingface’s transformers library ([Wolf et al., 2019](#))

Method	Macro-Avg F1
TF-IDF	0.61
WV	0.59
DB	0.73
DB-T	0.82

Table 3: Popularity Prediction: Macro-average F1 score for the baselines and the proposed Triplet training strategy for the popularity prediction task.

- **DISTILBERT (DB):** The embedding corresponding to the [CLS] token in a pre-trained DISTILBERT ([Sanh et al., 2019](#)) is taken as an embedding for a podcast.
- **DISTILBERT-Triplet (DB-T):** The embedding corresponding to the [CLS] token in a pre-trained DISTILBERT is trained in a Triplet manner as shown in the proposed method (Figure 3), and the output of the final neural network is a 128-dimensional embedding for the podcast.

For each of the methods listed above, embeddings corresponding to every podcast are extracted. We use a supervised classifier like XGBOOST ([Chen and Guestrin, 2016](#)) with binary labels for popularity. Results for the various methods are given in Table 3. Appropriate hyperparameter tuning is done over 5-fold cross validation, including adding penalties for misclassifying the minority (Popular) class. It can be seen that our proposed method (DB-T) significantly outperforms the others, achieving a relative improvement over the best baseline (DB) by 12.3%.³

6 Conclusion

In this work, we explore how textual cues like polarity, subjectivity, lexicons and keywords differ in popular and unpopular podcasts. We then employ a triplet-based training procedure to counter the class imbalance problem in our data, which yields a relative improvement of 12.3% over the best performing baseline. In future work, we plan to explore this problem in a multi-modal setting, by constructing multi-modal embeddings that leverage both audio and textual data. We also plan to leverage temporal information associated with the transcripts, in the form of timestamps of the spoken words, for the task of popularity prediction.

³Code and saved models are available at: <https://github.com/brihijoshi/podpop-nlp4musa-2020/>

References

- Tianqi Chen and Carlos Guestrin. 2016. [Xgboost](#). *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
- Ruth Dhanaraj and Beth Logan. 2005. Automatic prediction of hit songs. In *ISMIR*.
- Ethan Fast, Binbin Chen, and Michael S. Bernstein. 2016. [Empath](#). *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*.
- Elad Hoffer and Nir Ailon. 2015. Deep metric learning using triplet network. In *Similarity-Based Pattern Recognition*, pages 84–92, Cham. Springer International Publishing.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2, NIPS'13*, page 3111–3119, Red Hook, NY, USA. Curran Associates Inc.
- Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 133–142. New Jersey, USA.
- Abhishek Sanghi and Daniel G. Brown. 2014. Hit song detection using lyric features alone. In *Proceedings of the 15th International Society for Music Information Retrieval Conference 2014 (ISMIR 2014)*.
- Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. [Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter](#).
- Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. [Facenet: A unified embedding for face recognition and clustering](#). *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Yla R. Tausczik and James W. Pennebaker. 2010. [The psychological meaning of words: Liwc and computerized text analysis methods](#). *Journal of Language and Social Psychology*, 29(1):24–54.
- Manos Tsagkias, Martha Larson, and Maarten de Rijke. 2009. Exploiting surface features for the prediction of podcast preference. In *Advances in Information Retrieval*, pages 473–484, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Manos Tsagkias, Martha Larson, Wouter Weerkamp, and Maarten de Rijke. 2008. [Podcred: A framework for analyzing podcast preference](#). In *Proceedings of the 2nd ACM Workshop on Information Credibility on the Web, WICOW '08*, page 67–74, New York, NY, USA. Association for Computing Machinery.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R’emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface’s transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.
- L. Yang, S. Chou, J. Liu, Y. Yang, and Y. Chen. 2017. Revisiting the problem of audio-based hit song prediction using convolutional neural networks. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 621–625.
- Longqi Yang, Yu Wang, Drew Dunne, Michael Sobolev, Mor Naaman, and Deborah Estrin. 2019. [More than just words: Modeling non-textual characteristics of podcasts](#). In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM '19*, page 276–284, New York, NY, USA. Association for Computing Machinery.
- Eva Zangerle, Ramona Huber, and Michael Vötter Yi-Hsuan Yang. 2019. Hit song prediction: Leveraging low- and high-level audio features. In *Proceedings of the 20th International Society for Music Information Retrieval Conference 2019 (ISMIR 2019)*, pages 319–326.

Using Latent Semantics of Playlist Titles and Descriptions to Enhance Music Recommendations

Yun Hao and **J. Stephen Downie**

School of Information Sciences
University of Illinois at Urbana-Champaign
{yunhao2, jdownie}@illinois.edu

Abstract

Music playlists, either user-generated or curated by music streaming services, often come with titles and descriptions. While crucial to music recommendations, leveraging titles and descriptions is difficult due to sparsity and noise in the data. In this work, we propose to capture useful latent semantics behind playlist titles and descriptions through proper clustering of similar playlists. In particular, we clustered 20,065 playlists with both titles and descriptions into 562 groups using track vectors learned by word2vec model on over 1 million playlists. By fitting a Naive Bayes model on titles and descriptions to predict cluster membership and using the cluster membership information for music recommendations, we present a simple and promising solution to the cold-start problem in music recommendation. We believe that when combined with other sources of features such as audio and user interaction, the proposed approach would bring further enhancement to music recommendations.

1 Introduction

In the theory of Information Retrieval (IR), users formulate “queries” using natural language to express information needs to IR systems (Baeza-Yates and Ribeiro-Neto, 1999), and the query terms make up a sparse semantic space. Similarly, users on music streaming platforms, when creating new playlists, express their needs for music by providing playlist titles and descriptions. These playlist titles and descriptions also make up a highly sparse corpus, and capturing useful latent semantics from such sparse space for making music recommendations is challenging. In fact, using playlist titles for music recommendations can even worsen the performance of recommender systems (Zamani et al., 2018).

In this work, we present that through proper clustering of similar playlists, titles and descriptions can be effectively employed for making more accurate music recommendations. Specifically, two stages are included in this work: in Stage 1, track sequences in playlists are used to embed playlists and tracks into a latent embedding space using word2vec (Mikolov et al., 2013a), and agglomerative clustering is implemented on playlist embeddings to form clusters of similar playlists; in Stage 2, we fit a multinomial Naive Bayes model on words from playlist titles and descriptions to predict cluster membership and use the cluster membership information to make music recommendations. Details of the two stages are in Section 4 and Section 5, respectively. In Section 6, we evaluate the proposed recommending strategy by the task of making music recommendations given only playlist titles and descriptions, with several baseline models and strategies compared.

2 Related Work

Works have been done to capture hidden semantics from playlist titles for music recommendation. Pichl et al. (2015) formed clusters of playlist titles and interpreted each cluster as a latent music listening context for making music recommendations. The authors expanded the corpus by adding synonyms and hypernyms using WordNet (Miller, 1995) to deal with sparsity. The same authors later built on this work and formed situational clusters using selected playlist titles that contain activities and other descriptors (e.g., season, events) to improve music recommender systems (Pichl and Zangerle, 2018). One of the ACM RecSys Challenge 2018¹ tasks is to predict tracks in playlists given titles only. Approaches adopted

¹<http://www.recsyschallenge.com/2018/>



Figure 1: Example playlists from the datasets

by the top performing teams include matrix factorization on (playlist, track)-title co-occurrence matrix (Volkovs et al., 2018), character-level convolutional neural network to embed playlist titles (Yang et al., 2018), and using playlist titles as queries to pseudo-documents generated for each track by concatenating all the titles of the playlists that contained a particular track (Kallumadi et al., 2018).

Starting from the intuition that interpreting playlist titles and descriptions as plain text is not effective enough, we propose to fit a language model on titles and descriptions based on some “intermediate” information so that the “intermediate” information can guide us towards a better understanding of the language behind playlist generation.

3 Data

The datasets we used include the Million Playlist Dataset (MPD) released by Spotify for ACM RecSys Challenge 2018 as well as 1,417 playlists curated by Spotify collected via Spotify API². The MPD is further divided into two subsets, one with playlists with descriptions (D_1), and one with playlists without descriptions (D_2). Usage of each subset in this work will be detailed in later sections. To get more quality titles and descriptions data, we also collected 1,417 playlists curated by Spotify (D_3). Table 1 shows the summary of the datasets. Three example playlists from the datasets are shown in Figure 1. Playlist #1 is a curated playlist on Spotify, while playlist #2 and #3 are user-generated playlists that have been made public on Spotify.

4 Clustering of Playlists

4.1 Latent Representations of Playlists

Word embedding approaches, such as word2vec (Mikolov et al., 2013a) and GloVe (Pennington

²<https://developer.spotify.com/documentation/web-api/>

Dataset	Size
D_1 MPD w/ descriptions	18,760
D_2 MPD w/o descriptions	981,240
D_3 Spotify Curated Playlists	1,417

Table 1: Summary of the datasets

et al., 2014), provide an effective way to learn dense vector representations of words by leveraging word co-occurrence information. By treating playlists as the equivalent of sentences, and tracks as the equivalent of words, similar to (Kallumadi et al., 2018), we propose to apply word embedding approach to learn a dense vector representation for each of the unique track IDs and represent each playlist by aggregating its track embedding vectors. For learning the track embedding vectors, the word2vec model was chosen and the reasons are as follow: 1) with the continuous bag-of-words model of word2vec, ordering information is discarded and is more preferred in the setting of making “static” recommendations, as opposed to playlist continuation task in music listening session; 2) the linearity of the vector operations is claimed to weakly hold for the addition of several vectors by word2vec (Mikolov et al., 2013c), so aggregating track vectors should yield a meaningful representation of playlists.

All the 1,001,417 playlists in the dataset were used for learning the latent representations of playlists so that the learning process can make the most of the available data. In total, over 64 million unique tracks were fed to the word2vec model, and after subsampling (Mikolov et al., 2013b) we learned 50-dimensional latent representations of 600,501 tracks.

With the learned track vectors, each playlist in D_1 and D_3 (20,177 playlists in total) is represented as the average of its track vectors. Because not all tracks in the dataset has a dense vector, there are 112 playlists whose tracks are all absent from the latent embedding space. These playlists are discarded, leaving 20,065 playlists with descriptions in the dataset.

4.2 Clusters of Similar Playlists

With latent vector representations of playlists, groups of similar playlists can be formed using clustering algorithms. Among options such as K-means clustering and modularity-based clustering (Clauset et al., 2004), we found agglomerative clustering using cosine distances normalized for

#	Size	Top 10 words with highest BiTF from the cluster
1	628	oldies,80s,goodies,classics,soul,love,school,70s,dad,60s
2	597	rap,fire,hype,litty,chill,af,bangers,gang,trap,party
3	458	throwback,throwbacks,childhood,nostalgia,disney,2000s,school,tbt,middle school,bops
4	457	rock,classic,classics,classic rock,oldies,dad,roll,70s,80s,school
5	433	edm,house,electronic,dance,dubstep,chill,trap,gaming,bass,drops

Table 2: Top 5 largest clusters of similar playlists, presented by top 10 words from each cluster

each playlist yields the best result. In total, 580 clusters are formed in the data, including 18 singletons (clusters with 1 playlist). We removed the singletons to uncover general patterns in the data, leaving 562 clusters of similar playlists. Table 2 shows a summary of the top 5 largest clusters. From the table, it can be shown that playlists in the same cluster share something similar – genre, event, mood, etc..

5 Language Modeling on Playlist Titles and Descriptions via Naive Bayes

Given the clusters of playlists, we fit a multinomial Naive Bayes model using words from playlist titles and descriptions. Naive Bayes model was chosen because it is fast and accurate enough to serve as a proper baseline for text classification, and that with multinomial Naive Bayes, each cluster can be represented as a unigram language model which allows us to get more insights into the language used in playlist generation.

Before fitting the model, a series of data cleaning and preprocessing steps such as normalizing emojis was implemented on the text data. We omit the details for brevity here.

We chose binary term frequency (BiTF) as the text feature to extract from titles and descriptions because BiTF usually works better with short and sparse text. Bigrams were also included so that frequently mentioned artist names such as “Ed Sheeran” can be preserved. We further pruned the vocabulary with a minimum term frequency of 3, which yields a vocabulary of 5,487 tokens.

5.1 Model Details

Stratified sampling was implemented to split the playlists with descriptions (i.e., D_3) into training (19,044, 95%) and test set (1,003, 5%). We then fit a Naive Bayes model with Laplace smoothing on the training set.

6 Evaluation

6.1 Experimental Setup

Evaluation is done by the task of making music recommendations given playlist titles and descriptions. The baselines to compare are word2vec word embeddings trained on the training set, pre-trained GloVe word vectors on 2 billion tweets (both 50-dimensional and 200-dimensional), as well as the top-performing approach based on matrix factorization to dealing with cold-start playlists from the RecSys Challenge 2018 (v16³). Naive approaches that either recommend popular tracks or random tracks are also included as baselines.

For all approaches except v16 and the two naive approaches, one of two recommending strategies was employed according to the type of text features:

- Cluster-based: predict C potential clusters and recommend top tracks from the clusters by track frequencies weighted by normalized distances between the query playlist and the predicted clusters centers.
- Similarity-based: retrieve S similar playlists and recommend top tracks from the playlists by track frequencies weighted by normalized distances between the query playlist and the similar playlists.

We set $C = 5$ and $S = 5 \times 11 = 55$, where 11 is the median size of clusters in the training set, for fair comparison. Each model will return 500 candidates for evaluation.

F1, NDCG, R-precision, and R-artist are reported. F1 score measures the retrieval quality of the approaches while NDCG measures the ranking quality. R-artist is the same R-precision metric used for RecSys Challenge 2018 (Zamani et al., 2018), where artist matches were partially rewarded even if the predicted track was incorrect. For brevity, we only describe the standard

³<https://github.com/layer6ai-labs/RecSys2018/>

R-precision here. Let R be the set of ground truth tracks for a playlist, and T be the set of first $|R|$ tracks returned by the system. R-precision is then calculated as:

$$R\text{-precision} = \frac{|T \cap R|}{|R|} \quad (1)$$

6.2 Results and Analysis

Table 3 summarizes the evaluation results. Clearly, our proposed cluster-based strategy yields the most satisfying result. Of all the baseline approaches, similarity-based BiTF works the best, confirming that BiTF is a very effective text feature that works well for short and sparse text.

In the following two subsections, we present two examples to illustrate how the clusters may have helped with making more accurate recommendations.

Model	F1@100	F1@500	NDCG@100	NDCG@500	R-prec	R-artist
Cluster-based						
BiTF	0.0735	0.0524	0.0632	0.0652	0.0692	0.0717
Similarity-based						
BiTF	0.0713	0.0463	0.0623	0.0637	0.0663	0.0692
word2vec	0.0663	0.0432	0.0578	0.0589	0.0621	0.0641
GloVe-50d	0.0453	0.0309	0.0392	0.0400	0.0421	0.0436
GloVe-200d	0.0489	0.0339	0.0428	0.0438	0.0457	0.0472
Others						
v16	0.0661	0.0431	0.0588	0.0601	0.0624	0.0658
Popular	0.0381	0.0351	0.0308	0.0320	0.0337	0.0350
Random	0.0002	0.0003	0.0002	0.0002	0.0002	0.0003

Table 3: Evaluation results

6.3 Neighboring Clusters

One of the reasons why the clusters of similar playlists can help with making recommendations is that the clustering is effective to group similar playlists together. Figure 2 shows the 5 nearest neighbor clusters of the query cluster “Christmas” (shown in bold). According to the top words from each clusters, all the 5 neighbors seem to be relevant to “Christmas”; thus it is very likely that tracks from the neighboring clusters are good candidates to recommend given a query playlist comes from the “Christmas” cluster.

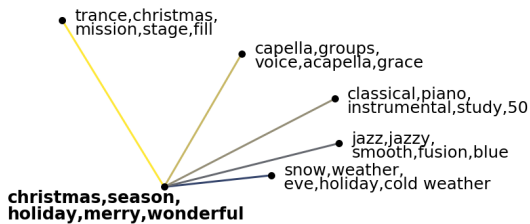


Figure 2: Neighboring clusters of “Christmas”. Edge lengths indicate distances from the query cluster to its neighbors.

6.4 Candidates with Diversity

By observing the behavior of the Naive Bayes model, it is interesting to see that when no additional information is provided to a query word, the Naive Bayes can recall more diverse potential candidates, which may benefit the recommender system. For example, in Figure 3 we show the top 5 most likely clusters from which word “study” is generated. Of the 5 candidate clusters, each indicates a different “group” or “genre” and each can be relevant to the query word “study” according to different user tastes or preferences – some people may prefer classical music or movie soundtracks when study, while some may prefer electronic dance music (edm) to stay energetic. When the recommender system has no additional knowledge about the user’s preference, it may be a better strategy to provide wider options for the user to choose from.

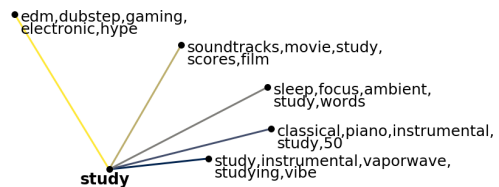


Figure 3: Top 5 clusters returned for query word “study”. A shorter edge length indicates a higher probability that the query belongs to the cluster.

7 Conclusion and Future Work

In this work, we present that through proper clustering of similar playlists, titles and descriptions can be effectively employed for making more accurate music recommendations. There are several future directions to extend this work. First, it is worth exploring how the method can be combined with audio signals and user interaction data to further benefit music recommender systems. Second, other aggregation of the track embeddings for playlists than averaging can be explored for making even more accurate recommendations. Lastly, evaluating the quality of music recommendations without user feedback data may not be accurate, especially when novelty and serendipity (Schedl et al., 2014) is preferred by users. Therefore, this work can benefit from some other datasets with feedback information available as ground truth.

References

- Ricardo A. Baeza-Yates and Berthier Ribeiro-Neto. 1999. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., USA.
- Aaron Clauset, Mark EJ Newman, and Christopher Moore. 2004. Finding community structure in very large networks. *Physical review E*, 70(6):066111.
- Surya Kallumadi, Bhaskar Mitra, and Tereza Iofciu. 2018. A line in the sand: Recommendation or ad-hoc retrieval? *arXiv preprint arXiv:1807.08061*.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.
- Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013c. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 746–751.
- George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.
- Martin Pichl and Eva Zangerle. 2018. [Latent Feature Combination for Multi-Context Music Recommendation](#). In *2018 International Conference on Content-Based Multimedia Indexing (CBMI)*, pages 1–6.
- Martin Pichl, Eva Zangerle, and Günther Specht. 2015. [Towards a Context-Aware Music Recommendation Approach: What is Hidden in the Playlist Name?](#) In *15th IEEE International Conference on Data Mining Workshops (ICDM 2015)*, ICDM 15, pages 1360–1365, Atlantic City. IEEE.
- Markus Schedl, Emilia Gómez Gutiérrez, and Julián Urbano. 2014. Music information retrieval: Recent developments and applications. *Foundations and Trends in Information Retrieval*. 2014 Sept 12; 8 (2-3): 127-261.
- Maksims Volkovs, Himanshu Rai, Zhaoyue Cheng, Ga Wu, Yichao Lu, and Scott Sanner. 2018. Two-stage model for automatic playlist continuation at scale. In *Proceedings of the ACM Recommender Systems Challenge 2018*, pages 1–6.
- Hojin Yang, Yoonki Jeong, Minjin Choi, and Jongwuk Lee. 2018. Mmcf: Multimodal collaborative filtering for automatic playlist continuation. In *Proceedings of the ACM Recommender Systems Challenge 2018*, pages 1–6.
- Hamed Zamani, Markus Schedl, Paul Lamere, and Ching-Wei Chen. 2018. An analysis of approaches taken in the acm recsys challenge 2018 for automatic music playlist continuation. *arXiv preprint arXiv:1810.01520*.

Prediction of User Listening Contexts for Music Playlists

Jeong Choi *
Knowledge AI Lab.
NCSOFT
jchoi@ncsoft.com

Anis Khlif
Deezer Research
research@deezer.com

Elena V. Epure
Deezer Research
research@deezer.com

Abstract

In this work, we set up a novel task of playlist context prediction. From a large playlist title corpus, we manually curate a subset of multi-lingual labels referring to user activities (e.g. ‘jogging’, ‘meditation’, ‘au calme’), which we further consider in the prediction task. We explore different approaches to calculate and aggregate track-level contextual semantic embeddings in order to represent a playlist and predict the playlist context from this representation. Our baseline results show that the task can be addressed with a simple framework using information from either audio or distributional similarity of tracks in terms of track-context co-occurrences.

1 Introduction

1.1 Motivation for user listening context prediction for playlists

The origination of playlists has changed over the last two decades. Before, it used to be regarded as the work of skilled DJs or curators who had significant musical knowledge and accessibility to music databases. However, as the general music consumption has shifted to streaming services and the entire music database has become accessible to anyone, the creation of playlists has become a common way for users to organise their music catalogue in coherent collections for different listening circumstances or with different themes (Pichl et al., 2016; Dias et al., 2017).

Hence, considering how pervasive playlists are in music streaming services, being able to automatically predict their possible listening contexts could enable us to perform context-aware track recommendation for playlist continuation or to generate context-centered playlist captions.

Track-level information, such as social tags, metadata and audio content, has been widely used

* This work is an extended version of the author’s research internship at Deezer.

Playlist titles (Deezer)	Track-level tags (Last.fm)
soiree , rock, <i>chill</i> , dance, cool, sport , pop, electro, divers, ambiance, party , funk, rap, running , love, annee 80, voiture , new, calme, <i>relax</i> , latino, gym , summer , house, oldies, classique, apero , mix, slow, musique	rock, pop, alternative, indie, electronic, female vocalists, favorites, Love, dance, 00s, alternative rock, jazz, beautiful, singer-songwriter, metal <i>chillout</i> , male vocalists, Awesome, classic rock, soul, indie rock, Mellow, electronica, 80s, folk, british, 90s, <i>chill</i> , american, instrumental

Table 1: 30 most commonly used titles from Deezer playlist dataset (left) and 30 most commonly used tags from Last.fm dataset (right). The bold text ones are related to ‘user-context’ category, and the normal ones are related to ‘music-context’ or ‘music-content’ categories. The italic ones could relate to either of ‘user-context’ or ‘music-content’.

in research efforts seeking to unveil the general musical semantics (Levy and Sandler, 2008; Nam et al., 2018) or context-related aspects (Ibrahim et al., 2020) of single tracks. However, to our knowledge, the problem of how to deduce the music listening context for playlists by relying on signals from their track constitution has not been yet researched.

1.2 Playlist titles as contextual cues

The word ‘context’ as employed by the recommendation system community encompasses a wide range of information such as activities, demographic information, emotional states, or weather-related information (Kaminskas and Ricci, 2012). In order to infer the user listening context, very diverse sources of data such as device logs are necessary (Cunningham et al., 2008; Wang et al., 2012; Gillhofer and Schedl, 2015), although in practical scenarios it is very challenging to access most of them while respecting user privacy.

The titles of user-created playlists, on the contrary, frequently encode information with regard to

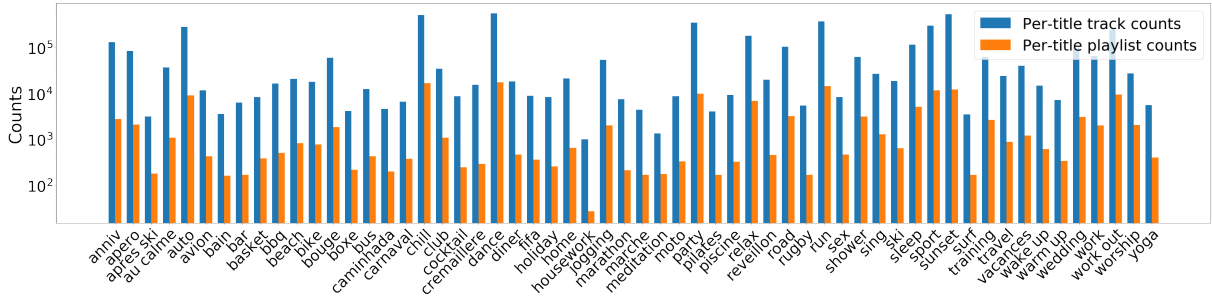


Figure 1: Per-title numbers of track instances (blue bar) / playlist instances (orange bar).

specific listening contexts and appear often as public user information (Pichl et al., 2015). These titles are noisy since they are crowd-sourced. However, a sufficiently large corpus can give statistically meaningful cues. In this work, we utilize a large playlist dataset where each playlist has associated a user-created title, which we leverage to infer the listening context that a playlist would fit.

Unlike the track-level tags or metadata, a playlist title is more likely to represent the user listening context of the corresponding sequence of music tracks. As shown in Table 1, among the 30 most commonly used playlist titles in the Deezer playlist dataset, 8 are related to *user-context* category rather than to *music-context* or *music-content* categories (Schedl, 2013) compared to none in the track-level tags dataset (Last.fm).

While there have been multiple research works that leverage playlist titles as supplementary information for a music recommendation or playlist continuation task (Pichl et al., 2015; Zamani et al., 2019), the playlist title prediction task has not been studied. In the current work, we focus on a subset of titles referring to the context. However, the method we explore could be easily adapted to new title categories.

1.3 Context-related title prediction for playlists

The largely overlapping track-level information between different playlist titles can pose difficulties for the playlist title prediction task. For example, tracks in a playlist with the title ‘*running*’ might be very similar to ones in a playlist with the title ‘*workout*’ (Ibrahim et al., 2020). A previous work (Pichl et al., 2015) has tried clustering playlists with lemmatized titles to use as an additional feature for the recommendation system, while another research work (McFee and Lanckriet, 2012) has attempted to tackle this overlapping characteristic issue with a hypergraph model.

However, the explicit distinction between different playlist contexts is left unclear even though those past works have helped improving the recommendation performance.

Here, we propose a framework to extract a semantic representation of a playlist as a low dimensional embedding related to its title or a specific desired concept such as the context, or user activities. To evaluate the representational power of these embeddings, we design and conduct activity-related title prediction experiments and compare the results obtained with different architectures.

2 Data preparation for user activity prediction from playlists

To set up a playlist dataset with activity labels, we first collected 2M user-created playlists from Deezer along with their titles. After a text cleaning and normalization¹ procedure, we chose 1,000 most commonly used playlist titles as our initial candidates.

A manual annotation experiment was further organised. Three music information retrieval researchers annotated each title as corresponding to a specific user activity or not. Then, 176 titles that were voted by at least two out of three annotators were selected (majority voting). Since Deezer playlist titles were multi-lingual, we further merged some cross-lingual synonyms into a single representative label, ending up with 58 activity categories (see Figure 1). We split the playlists into training (80%) and test set (20%) in stratified way, and filtered out any tracks that occur only on the test set playlists. This is because one of our baseline approaches requires track-level embeddings computed from the track-title matrix of the training set. The whole procedure left us with

¹We lowercase and remove special characters, although we keep emoji’s and some of widely used combinations of special characters manually chosen. (e.g. ‘<3’ or ‘:’)’)

156,269 playlists that had one of the 58 activity-related titles and 154,611 unique tracks included in these playlists. The average number of tracks in a playlist was 46.38 and their standard deviation was 36.08.

3 Baseline playlist embedding models

Playlist embedding task is a many-to-one inference problem where sequential data inputs are aggregated to infer one embedding, in this case context-related. This problem is similar to the sentence embedding problem from the natural language processing field. Tracks are constitutive elements of a playlist as words are of a sentence (Kalchbrenner et al., 2014).

3.1 Using title-track matrix factorization (MF) based embeddings

Our first approach is to apply a 2-step procedure. We first compute track-level semantic embeddings based on title annotations in the playlist corpus. Then, for a given playlist, we aggregate all the track-level embeddings to make a sequence-level prediction (detailed in Section 3.3)

We aim to extract track-level embeddings that represent the ‘distributional similarity’ of tracks. That is, the embeddings of tracks that occur together more often (are similarly distributed) within playlists with the same title will be trained to be closer. This is a basic strategy to learn word embeddings and train such semantic models in the natural language processing field (Mikolov et al., 2013; Pennington et al., 2014).

By seeing a playlist as a sentence and a track as a word, we can apply any of widely used modelling techniques that extract the semantic (thematic) embedding of each track, such as Latent Dirichlet Allocation, Skip-gram (implicit matrix factorization), Word2vec, GloVe etc. (Blei et al., 2003; Levy and Goldberg, 2014; Mikolov et al., 2013; Pennington et al., 2014). Another option is to construct a matrix of playlist titles and track counts to conduct singular value decomposition or matrix factorization, and thus get an embedding for each track (Sarwar et al., 2001; Zhou et al., 2008; Hu et al., 2008).

Among these options, we chose the matrix factorization that allowed the extraction of track embeddings along with title embeddings simultaneously. We used the playlists in the training set to construct the ‘title-by-track co-occurrence’ matrix

by adding up all track counts from playlists that are annotated with the same title. We then normalized the matrix track-wise after computing TF-IDF values. We applied alternating least square algorithm (Bell and Koren, 2007) to factorize the matrix, resulting in a 50-dim feature vector for each track and title.

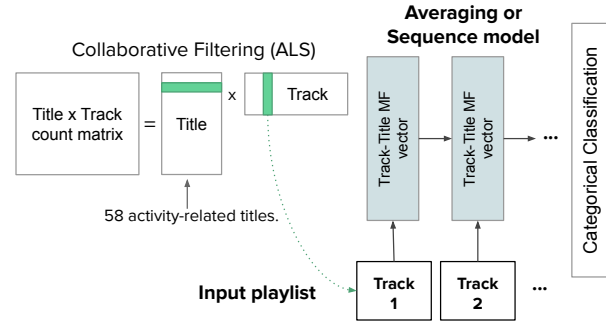


Figure 2: Diagram of MF-based embedding model.

3.2 Using audio-based embeddings

Our second approach is to learn track embeddings directly from the audio content. We set up a CNN architecture using a mel-spectrogram input that were computed with 22,050 Hz sampling rate, 1,024 FFT size, 512 hop size, and 128 mel bins. A 3-second long mel-spectrogram segment is put into the network with 5 layers of 1D convolution. The network outputs 50-dim feature vector for each segment, and we average them to end up with a 50-dim embedding for each track. In this case, track-level audio embeddings are jointly trained with the aggregated playlist embeddings in an end-to-end manner.

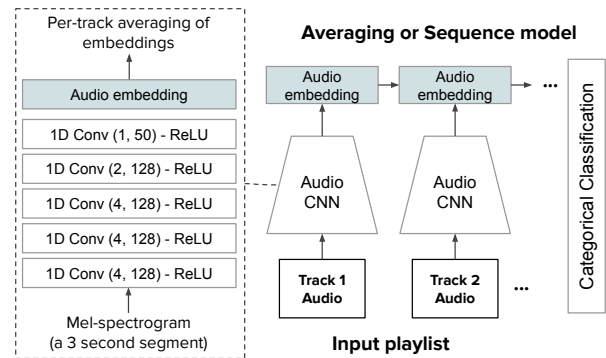


Figure 3: Diagram of audio embedding model.

3.3 Aggregation techniques of track embeddings into playlist embedding

The aggregation of track embeddings into a playlist representation is done in two ways: one is

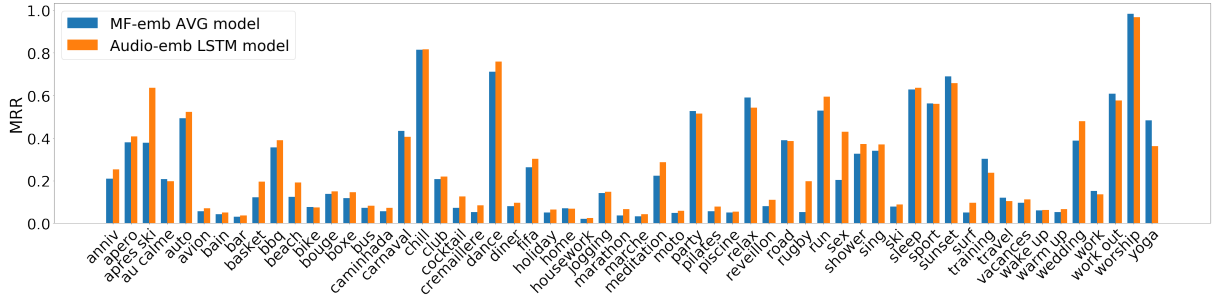


Figure 4: MRR results per each title (MF embedding averaging model performances (blue bar) / audio embedding LSTM model performances (orange bar)).

Model	MRR	FH@1	FH@5	MAP@5
MF-emb_AVG	0.532	0.358	0.758	0.509
MF-emb_LSTM	0.516	0.341	0.744	0.492
Audio_AVG	0.533	0.359	0.759	0.510
Audio_LSTM	0.543	0.371	0.771	0.521

Table 2: Baseline results on the playlist activity prediction task. *MF-emb* denotes the matrix factorization based embedding model, and *Audio* denotes the audio-based embedding model. (MRR : mean reciprocal rank / FH:flat hit / MAP:mean average precision)

by simply averaging track-level embeddings and the other is by using a single-layered LSTM network that takes a sequence of track embeddings as an input. After computing the aggregated information for a single playlist, the resulted playlist embedding is used as an input to a fully connected layer and a softmax layer that outputs the prediction for one of the 58 activity labels.

4 Results and discussion

As shown in Table 2, models using audio-based embeddings performed slightly better than the ones using the MF-based embeddings. One interesting finding was that, for models using the MF-based embeddings, the model was very prone to overfit to the training set. This could be because the track-level input embeddings were computed from the matrix that partly originated from the playlist-title table that the model was being trained to predict. In this case, the simple approach of averaging track embeddings ended up performing better than making use of track-level details or the sequential information. On the other hand, for models using the audio-based embeddings, a more complex architecture that considers track-level details and the sequential order performed better, as expected.

Investigating the prediction performance on each title (see Figure 4), ‘worship’, ‘chill’, ‘dance’, and ‘sunset’ were the most accurately

	Num. of tracks per title	Num. of playlists per title
MRR per title (MF-emb_AVG)	0.7137	0.7472
MRR per title (Audio.LSTM)	0.6881	0.7227

Table 3: Pearson correlation coefficients between our models’ prediction performances (MRR) and the numbers of instances for each title.

predicted ones for all the models. However, we are facing a class imbalance problem where models are misguided to predict a title of the largest sample size when playlists with different titles have similar sequences of tracks. For example, for playlists labeled with ‘caminhada (walk)’, ‘marathon’, or ‘joggin’, models would be trained to predict as ‘run’ to simply achieve higher overall accuracy. As shown in Table 3, the sample size and the accuracy per title have a meaningful correlation.

Comparing results from different input representations, ‘apres ski’, ‘sex’, and ‘wedding’ were more accurately predicted by the audio-based embedding models, while ‘yoga’ and ‘training’ were more accurately predicted by the MF embedding-based models.

Our initial results show that there is a large room to discover about how each playlist is constructed for different user listening contexts. For the top-1 prediction, almost half of the activity titles could not be predicted correctly even for a single playlist. For the future work, we plan to improve the selection of the representative context titles, handle the class imbalance problem, and experiment more advanced architectures, such as self-attention architectures, to aggregate the track-level sequential information. A multi-modal approach combining the two input representations along with any extra information such as lyrics, track metadata or user embeddings could also be promising.

References

- Robert M Bell and Yehuda Koren. 2007. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 43–52. IEEE.
- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3:993–1022.
- Stuart Cunningham, Stephen Caulder, and Vic Grout. 2008. Saturday night or fever? context-aware music playlists. *Proc. Audio Mostly*.
- Ricardo Dias, Daniel Gonçalves, and Manuel J Fonseca. 2017. From manual to assisted playlist creation: a survey. *Multimedia Tools and Applications*, 76(12):14375–14403.
- Michael Gillhofer and Markus Schedl. 2015. Iron maiden while jogging, debussy for dinner? In *International Conference on Multimedia Modeling*, pages 380–391. Springer.
- Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*, pages 263–272. Ieee.
- Karim M Ibrahim, Jimena Royo-Letelier, Elena V Epure, Geoffroy Peeters, and Gaël Richard. 2020. Audio-based auto-tagging with contextual tags for music. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 16–20. IEEE.
- Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 655–665, Baltimore, Maryland. Association for Computational Linguistics.
- Marius Kaminskis and Francesco Ricci. 2012. Contextual music information retrieval and recommendation: State of the art and challenges. *Computer Science Review*, 6(2-3):89–119.
- Mark Levy and Mark Sandler. 2008. Learning latent semantic models for music from social tags. *Journal of New Music Research*, 37(2):137–150.
- Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185.
- Brian McFee and Gert RG Lanckriet. 2012. Hypergraph models of playlist dialects. In *ISMIR*, volume 12, pages 343–348. Citeseer.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.
- Juhan Nam, Keunwoo Choi, Jongpil Lee, Szu-Yu Chou, and Yi-Hsuan Yang. 2018. Deep learning for audio-based music classification and tagging: Teaching computers to distinguish rock from bach. *IEEE signal processing magazine*, 36(1):41–51.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.
- Martin Pichl, Eva Zangerle, and Günther Specht. 2015. Towards a context-aware music recommendation approach: What is hidden in the playlist name? In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 1360–1365. IEEE.
- Martin Pichl, Eva Zangerle, and Günther Specht. 2016. Understanding playlist creation on music streaming platforms. In *2016 IEEE International Symposium on Multimedia (ISM)*, pages 475–480. IEEE.
- Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. [Item-based collaborative filtering recommendation algorithms](#). In *Proceedings of the 10th International Conference on World Wide Web, WWW '01*, page 285–295, New York, NY, USA. Association for Computing Machinery.
- Markus Schedl. 2013. Ameliorating music recommendation: Integrating music content, music context, and user context for improved music retrieval and recommendation. In *Proceedings of international conference on advances in mobile computing & multimedia*, pages 3–9.
- Xinxi Wang, David Rosenblum, and Ye Wang. 2012. Context-aware mobile music recommendation for daily activities. In *Proceedings of the 20th ACM international conference on Multimedia*, pages 99–108.
- Hamed Zamani, Markus Schedl, Paul Lamere, and Ching-Wei Chen. 2019. An analysis of approaches taken in the acm recsys challenge 2018 for automatic music playlist continuation. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(5):1–21.
- Yunhong Zhou, Dennis Wilkinson, Robert Schreiber, and Rong Pan. 2008. Large-scale parallel collaborative filtering for the netflix prize. In *International conference on algorithmic applications in management*, pages 337–348. Springer.

An Information-based Model for Writing Style Analysis of Lyrics

Melesio Crespo-Sanchez¹, Edwin Aldana-Bobadilla², Ivan Lopez-Arevalo¹, and Alejandro Molina-Villegas³

¹{melesio.crespo, ilopez}@cinvestav.mx

²edwyn.aldana@cinvestav.mx

³amolina@centrogeo.edu.mx

¹Cinvestav - Tamaulipas, Victoria, Mexico.

²Conacyt - Cinvestav, Victoria, Mexico.

³Conacyt - Centro de Investigación en Ciencias de Información Geoespacial, Mexico.

Abstract

One of the most important parts of the song’s content is the lyrics, in which authors expose feelings or thoughts that may reflect their way of seeing the world. This is perhaps the reason why modern techniques of mining text have been applied to lyrics to find semantic aspects that allow us to recognize emotions, topics, authorship among others. In this work, we focus on the analysis of syntactic aspects assuming that they are important elements to recognize patterns related to the writing style of an individual author or a musical genre. We present a theoretical information model-based in a corpus of lyrics, which allows finding discriminating elements in a writing style that could be used to estimate, for example, the authorship or musical genre of a given lyric.

1 Introduction

Text mining has been applied to the analysis of lyric content in recent years to extract valuable hidden information. Since this content is not directly amenable to numerical computation, a *feature engineering process* is applied to extract features from the text. This process may include word embeddings (Espinosa-Anke et al., 2017) or probabilistic models (McFee and Lanckriet, 2011). From a set of numerical features, it is possible to create computational models to recognize patterns associated with the content of the lyrics. This recognition allows to carry out automate tasks such as topic modeling (Devi and Saharia, 2020), semantically similar lyrics detection (Chandra et al., 2020), sentiment analysis (Akella and Moh, 2019), text summarizing (Fell et al., 2019), automatic lyric generation (Potash et al., 2015), linguistic analysis (Petrie et al., 2008), explicit content detection (Chin et al., 2018) and music recommendation systems (Dong et al., 2020). Typically, these models highlight semantic aspects associated with the content of lyrics, leaving aside

other important aspects such as those associated with the way the content is written. In this regard, we propose a method that considers syntactic aspects to recognize different writing styles in song lyrics.

This work is organized as follows: In Section 2, we present the underlying ideas that support the main line of our proposal. In Section 3, we present the assessment methodology to determine the effectiveness of our proposal. In Section 4, we show obtained results from experiments. Finally, in Section 5, we present some brief conclusions.

2 Background

One important aspect when we have a text is to quantify the information in it. Where important elements of text must be identified, we considered the following concepts.

2.1 Information modeling

Information in a text can be defined as the facts about a situation, person, idea, among others, that are part of a document that follows certain grammatical and vocabulary rules in any language. This information can be modeled in the next ways:

- *Information Content*. Given a random variable Y , the information of the event ($Y = y_i$) is inversely proportional to its likelihood. This information is denoted by $I(y_i)$ and expressed as (Shannon, 1948):

$$I(y_i) = \log\left(\frac{1}{p(y_i)}\right) = -\log(p(y_i)) \quad (1)$$

For example, is the word love very common in lyrics? the informative content (I love) is expected to be low compared to other words less likely.

- *Shannon Entropy*. The expected value of I is known as Shannon’s Entropy, which is de-

defined as (Shannon, 1948):

$$H(Y) = - \sum_{i=1}^N p(y_i) \log(p(y_i)) \quad (2)$$

When $p(y_i)$ is uniformly distributed, the entropy of Y is maximal. This means that Y has the highest level of unpredictability and, thus, the maximal information content. Regarding the lyrics, the entropy rises when songs are more heterogeneous (in terms of grammar diversity).

2.2 POS Tagging

A key task in the text analysis is the Part Of Speech Tagging (POS Tagging). The POS of a word is its grammatical category associated. From the categories, it is possible to find grammatical structures and recognize elements related to things, ideas, people, etc. relevant to the information in the text. The POS Tagging is the process in which words are marked in a document as their corresponding POS category, based on its definition and context. A word strongly depends on its context and may have different categories depending on it (Toutanova et al., 2004).

3 Proposal

Our proposal aims to model the writing style of artists from a corpus of lyrics. In a corpus of lyrics grouped by artist, we encoded for each artist, the lyrics as a set of codes that represent syntactic structures based on POS (see Section 2.2). These codes are denoted as a discrete random variable Y_i . The distribution of Y_i is approximated through the frequency of the codes in the artist’s lyrics. Based on this distribution, the entropy $H(Y_i)$ can be computed. We hypothesize that these values could represent a measure of *the diversity of grammatical structures* contained in the discourse of the lyrics. Based on these ideas, we propose a method that follows the pipeline illustrated in Figure 1.

As mentioned, we have a song lyrics corpus description shown in Section 4. We analyze the way artists use syntactic structures in songs to determine how are written, we focus on the POS tags used and their combinations (syntactic structures). The description of each process is described as follows:

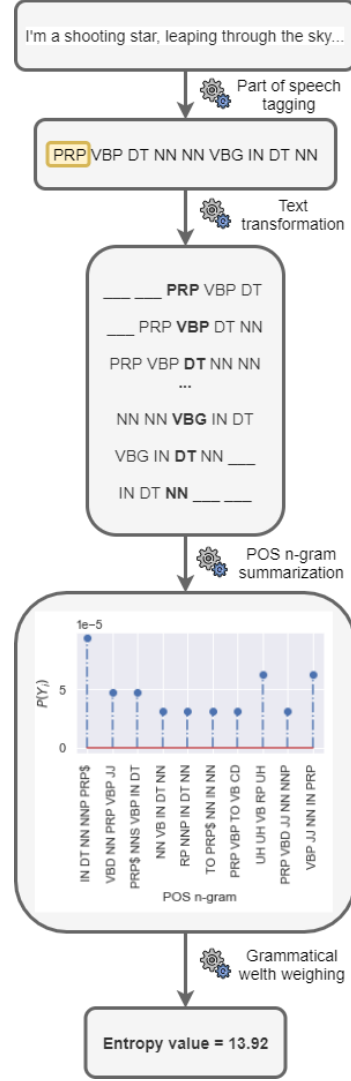


Figure 1: Proposed method to obtain an Information-based Model for Writing Style Analysis of Lyrics.

1. **Pre-processing.** Punctuation symbols, line breaks, and extra spaces removal are the pre-processing tasks in this process.
2. **Text transformation.** We extract the syntactical structures from the lyrics as follows: From the original text, a list L of POS tags is obtained. For each $l_i \in L$, we define a POS n -gram as a concatenation of l_i and its surrounding POS tags, expressed as: $l_{i-w} + \dots + l_{i-1} + l_i + l_{i+1} + \dots + l_{i+w}$, where w is the number of left and right POS tags, implying a length of the n -gram of $n = 2w + 1$. The above concatenation represents what we call *grammatical structures* used by an artist. When there are no POS tags to the left or right of l_i , we fill these spaces with underscores ($_$), this happens when l_i is either at the beginning

or the end of the lyrics.

3. **POS n-gram summarization.** POS n-grams can be seen as symbols used in the lyrics which stands for the random variable Y defined in Section 2.1. To summarize the syntactic structures used by the artists, we take all the POS n-grams previously obtained and model them as a probability distribution by the artist.
4. **Grammatical wealth weighing.** We intend to determine the writing style of the lyrics performed calculating the entropy of the POS n -grams distributions through Shannon’s entropy function defined in Equation 2. We used \log_2 in this paper. By performing this operation, we measure the variety of POS n-grams used in the lyrics. As a result, we could say that the grammatical wealth in lyrics is being weighed, where such entropy abstracts the artists’ writing style.

4 Results

In the experiments, we used the 55000+ Song’s Lyrics corpus obtained from the Kaggle repository¹. This corpus is composed of around 55 thousand instances of English written lyrics with the next features: *Artist*, *Song*, *Link*, and *Text*.

We used a value of $w = 2$ in this experiment. Since there were artists with very few lyrics on the corpus, only instances from artists with more than 100 lyrics were taken into account. For the POS tagging task we used the Stanford POS Tagger (Toutanova et al., 2003), which is reported to have a token accuracy of 97.24%. As result, a total of 268 different artists were selected from the corpus, whose entropy was calculated via the proposed method and ranked in descending order. We obtained the top and bottom entropy values by artists such as are shown in Table 1 and Table 2 respectively. The full results can be found on a web repository that we refer to in what follows as *experimental repository*².

The top ten artists shown in Table 1 are filled by Hip Hop or rap artists which are well known to use complex grammar structures as well as a diversity of word combinations (a common feature in this music genre). It is worth noting that in the same order of entropy is Bob Dylan, who won the Nobel

¹<https://www.kaggle.com/datasets>

²http://bit.do/entropy_lyrics

Rank	Artist	Entropy	Genre
1	LL Cool J	14.5948	Hip Hop
2	Insane Clown Posse	14.5056	Rap
3	Lil Wayne	14.4834	Hip Hop
4	Fabulous	14.4310	Hip Hop
5	Drake	14.3708	Hip Hop
6	R. Kelly	14.2982	Hip Hop
7	Kanye West	14.2623	Hip Hop
8	Bob Dylan	14.2475	Folk
9	Indigo Girls	14.1750	Rock
10	Joni Mitchell	14.1160	Jazz

Table 1: Top 10 artists ranked per entropy values, which denote the highest grammatical diversity.

Rank	Artist	Entropy	Genre
259	Warren Zevon	13.0663	Rock
260	Norah Jones	13.0592	Jazz
261	Wishbone Ash	13.0533	Rock
262	Whitesnake	13.0485	Rock
263	Regine Velasquez	13.0433	Pop
264	Misfits	13.0151	Punk
265	Steve Miller Band	12.9785	Rock
266	Yngwie Malmsteen	12.9450	Metal
267	Planetshakers	12.6079	Christian
268	Nirvana	12.4815	Rock

Table 2: Bottom 10 artists ranked per entropy values, which denote the lowest grammatical diversity.

Prize in Literature 2016 awarded “*for having created new poetic expressions within the great American song tradition*”. Opposed to the artists using complex grammar structures is the bottom ten artists shown in Table 2 which are characterized by using simple grammar structures.

The above can be summarized in Figure 3 wherein is shown the distribution of the entropy values per artist. The tails of the distribution contain the lower and higher entropy values, corresponding to the top and bottom artists previously shown. We can argue that an artist with a higher value of entropy, exhibits a greater diversity of POS n-grams in its lyrics than artists whose entropy is lower. This diversity, in terms of entropy, is an interesting finding that could reflect the grammatical wealth of the artists’ lyrics.

Notice that the entropy values follow a normal distribution wherein the most likely values (around the mean) would correspond to artists

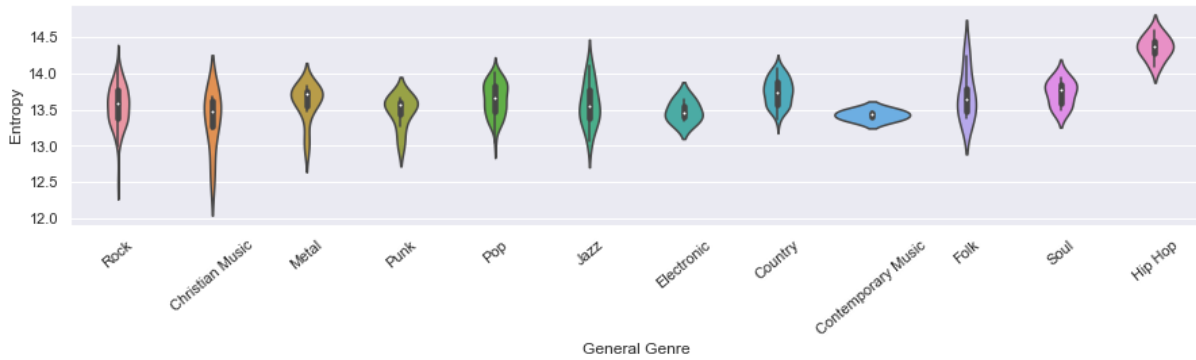


Figure 2: Artists' entropy distributions per music genre.

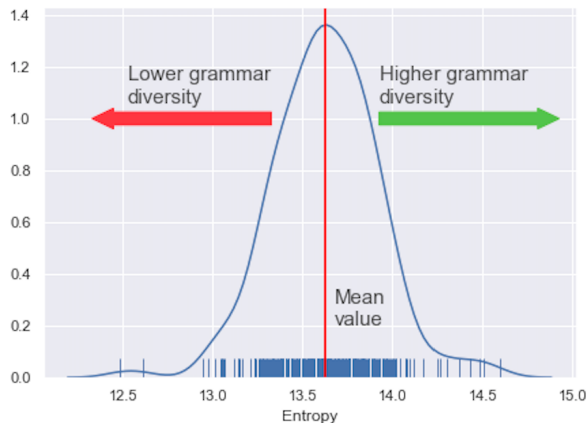


Figure 3: Artists' entropy distribution.

with an ordinary writing style (in grammar diversity terms). We are interested in the tails of the distribution, where we can find the lowest and the highest diversity. In this regard, we have included in the *experimental repository*, a comparison of two lyrics corresponding to the artist with the lowest and highest entropy (*Nirvana* and *LL Cool J* respectively). We can see an important difference between these lyrics from what we have called grammar diversity.

We conducted another experiment to remark the entropy distribution per music genre. In this case, we manually labeled the 268 artists where the results of these distributions are shown in Figure 2 represents the distribution of the artists' entropy values by genre. Here the differences between genres can be appreciated in the shape of the distributions.

The variance between distributions also tells us some differences between genres. Comparing the *Rock* and *Electronic* genres. In the first one, we can find lyrics with a very low or very high diversity of POS n-grams, given the variance of the dis-

tribution. Furthermore, the language in electronic music tends to be simple because this genre tends to focus more on music than on lyrics, this effect can be resumed in the variance of its entropy distribution, which is lower than in the first case.

The entropy in music genres can be different even if they have similar variance in their distributions, such as the case of *Country* and *Hip Hop*. In this case, Hip Hop lyrics tend to have higher values of entropy than Country lyrics since they use more POS n-grams to make rhymes.

5 Conclusions

In this work, we have analyzed how artists make use of grammatical structures trying to identify the writing style in their lyrics (syntactical approach) rather than the meaning of words in them (semantic approach). By abstracting the syntactical structures that are used in the lyrics with an entropy value, we have found that the writing style in lyrics tends to approximate a normal distribution which can be the result of the common syntactical structures used in the English language. Nevertheless, remarkable observations were found in certain artists' writing style and also when analyzing entropy by music genre. We intend that the results obtained in this analysis can be used to develop a new representation that refers to lexical, semantic, and syntactic elements in the abstraction of the text.

References

- Revanth Akella and Teng-Sheng Moh. 2019. Mood classification with lyrics and convnets. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 511–514. IEEE.

- J. Chandra, Akshay Santhanam, and Alwin Joseph. 2020. Artificial intelligence based semantic text similarity for rap lyrics. In *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, pages 1–5. IEEE.
- Hyojin Chin, Jayong Kim, Yoonjong Kim, Jinseop Shin, and Mun Y Yi. 2018. Explicit content detection in music lyrics using machine learning. In *2018 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 517–521. IEEE.
- Maibam Debina Devi and Navanath Saharia. 2020. Exploiting topic modelling to classify sentiment from lyrics. In *International Conference on Machine Learning, Image Processing, Network Security and Data Sciences*, pages 411–423. Springer.
- Yuchen Dong, Xiaotong Guo, and Yuchen Gu. 2020. Music recommendation system based on fusion deep learning models. In *Journal of Physics: Conference Series*, volume 1544, page 012029. IOP Publishing.
- Luis Espinosa-Anke, Sergio Oramas, Horacio Saggion, and Xavier Serra. 2017. Elmdist: A vector space model with words and musicbrainz entities. In *European Semantic Web Conference*, pages 355–366. Springer.
- Michael Fell, Elena Cabrio, Fabien Gandon, and Alain Giboin. 2019. Song lyrics summarization inspired by audio thumbnailing.
- Brian McFee and Gert RG Lanckriet. 2011. The natural language of playlists. In *ISMIR*, volume 11, pages 537–541.
- Keith J Petrie, James W Pennebaker, and Borge Sivertsen. 2008. Things we said today: A linguistic analysis of the beatles. *Psychology of Aesthetics, Creativity, and the Arts*, 2(4):197.
- 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.
- Claude Elwood Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.
- Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2004. [Feature-rich part-of-speech tagging with a cyclic dependency network](#). *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology—NAACL '03*, 1.
- Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 conference of the North*
- American chapter of the association for computational linguistics on human language technology—volume 1*, pages 173–180. Association for Computational Linguistics.

Generation of lyrics lines conditioned on music audio clips

Olga Vechtomova, Gaurav Sahu, Dhruv Kumar

University of Waterloo

{ovechtom, gsahu, d35kumar}@uwaterloo.ca

Abstract

We present a system for generating novel lyrics lines conditioned on music audio. A bimodal neural network model learns to generate lines conditioned on any given short audio clip. The model consists of a spectrogram variational autoencoder (VAE) and a text VAE. Both automatic and human evaluations demonstrate effectiveness of our model in generating lines that have an emotional impact matching a given audio clip. The system is intended to serve as a creativity tool for songwriters.

1 Introduction

Creative text synthesized by neural text generative models can serve as inspiration for artists and songwriters when they work on song lyrics. Novel and unusual expressions and combinations of words in generated lines can spark an idea and inspire the songwriter to create original compositions. In contrast to systems that generate lyrics for an entire song, our system generates suggestions in the form of individual lyrics lines, and is intended to serve as a creativity tool for artists, rather than as a standalone songwriting AI system.

In a song, musical composition, instrumentation and lyrics act together to express the unique style of an artist, and create the intended emotional impact on the listener. Therefore, it is important that a lyrics generative model takes into account the music audio in addition to the textual content.

In this paper we describe a bimodal neural network model that uses music audio and text modalities for lyrics generation. The model (Figure 1) generates lines that are conditioned on a given music audio clip. The intended use is for an artist to play live or provide a pre-recorded audio clip to the system, which generates lines that match the musical style and have an emotional impact matching the given music piece.

The model uses the VAE architecture to learn latent representations of audio clips spectrograms. The learned latent representations from the spectrogram-VAE are then used to condition the decoder of the text-VAE that generates lyrics lines for a given music piece. Variational autoencoder lends itself very well for creative text generation applications, such as lyrics generation. It learns a latent variable model of the training dataset, and once trained any number of novel and original lines can be generated by sampling from the learned latent space.

Three main groups of approaches towards stylized text generation in natural language processing (NLP) include: (1) embedding-based techniques that capture the style information by real-valued vectors, and can be used to condition a language model (Tikhonov and Yamshchikov, 2018) or concatenated with the input to a decoder (Li et al., 2016); (2) approaches that structure latent space to encode both style and content, and include Gaussian Mixture Model Variational Autoencoders (GMM-VAE) (Shi et al., 2020; Wang et al., 2019), Conditional Variational Autoencoders (CVAE) (Yang et al., 2018), and Adversarially Regularized Autoencoders (ARAE) (Li et al., 2020); (3) approaches with multiple style-specific decoders (Chen et al., 2019).

All of the above papers infer style from only one modality, text. Our work belongs to the first category of approaches: embedding based techniques, and is different from all of the above works in learning the style information from audio and text. Furthermore, previous embedding-based approaches use embeddings from a discrete vector space. This is the first work that uses a continuous latent variable learned by a spectrogram-VAE as a conditioning signal for the text-VAE.

A number of approaches have been proposed towards poetry generation, some focusing on

rhyme and poetic meter (Zhang and Lapata, 2014), while others on stylistic attributes (Tikhonov and Yamshchikov, 2018). Cheng et al. (2018) proposed image-inspired poetry generation. Yang et al. (2018) generated poetry conditioned on specific keywords. Yu et al. (2019) used audio data for lyrics retrieval. Watanabe et al. (2018) developed a language model for lyrics generation using MIDI data. Vechtomova et al. (2018) generated author-stylized lyrics using audio-derived embeddings. To our knowledge this is the first work that uses audio-derived data to generate lyrics for a given music clip.

The main contributions of this work are: (1) A probabilistic neural network model for generating lyrics lines matching the musical style of a given audio clip; (2) We demonstrate that continuous latent variables learned by a spectrogram-VAE can be effectively used to condition text generation; (3) Automatic and human evaluations show that the model can be effectively used to generate lyrics lines that are consistent with the emotional effect of a given music audio clip.

2 Background: unconditioned text generation with VAE

The variational autoencoder (Kingma and Welling, 2014) is a stochastic neural generative model that consists of an encoder-decoder architecture. The encoder transforms the input sequence of words x into the approximate posterior distribution $q_\phi(z|x)$ learned by optimizing parameters ϕ of the encoder. The decoder reconstructs x from the latent variable z , sampled from $q_\phi(z|x)$. Both encoder and the decoder in our work are recurrent neural networks, specifically, Long Short Term Memory networks (LSTM). The reconstruction loss is the expected negative log-likelihood of data:

$$J_{\text{rec}}(\phi, \theta, x) = - \sum_{t=1}^n \log p(x_t | z, x_1 \dots x_{t-1}) \quad (1)$$

where ϕ and θ are parameters of the encoder and decoder, respectively. The overall VAE loss is

$$J = J_{\text{rec}}(\phi, \theta, x) + \text{KL}(q_\phi(z|x) || p(z)) \quad (2)$$

where the first term is the reconstruction loss and the second term is the KL-divergence between z 's posterior and a prior distribution, which is typically set to standard normal $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

3 Approach

The audio clip conditioned generation model (Figure 2) consists of a spectrogram-VAE to learn a meaningful representation of the input spectrogram, and a text-VAE with an audio-conditioned decoder to generate lyrics.

In order to train the spectrogram-VAE, we first split the waveform audio of songs into small clips, and transform them into MEL spectrograms. For this mode of generation, we generate data with two levels of audio-lyrics alignment: high-precision and low-precision, which are described in more detail in Section 4. The spectrogram-VAE follows an encoder-decoder architecture. The encoder consists of four convolutional layers followed by a fully connected layer, and the decoder architecture is mirrored by using a fully-connected layer followed by four deconvolutional layers. This model was trained for 100 epochs.

We then feed all the spectrograms in the dataset to obtain their respective spectrogram embeddings. More precisely, we first obtain the μ and σ for every data point, and then sample a latent vector from the learned posterior distribution using a random normal noise $\epsilon \in \mathcal{N}(\mathbf{0}, \mathbf{I})$.

The audio clip conditioned VAE, which has the same architecture as described in Section 2, is then trained to generate lyrics befitting the provided piece of music by concatenating the spectrogram embedding with the input to every step of the decoder. The reconstruction loss is calculated as:

$$J_{\text{rec}}(\phi, \theta, z_k^{(s)}, x^{(t)}) = - \sum_{i=1}^n \log p(x_i^{(t)} | z_k^{(s)}, z_k^{(s)}, x_1^{(t)} \dots x_{i-1}^{(t)}) \quad (3)$$

where $z_k^{(s)}$, spectrogram embedding of the k -th data point. At inference time, a latent vector $z_k^{(t)}$ sampled from the text-VAE's prior is concatenated with the corresponding spectrogram embedding $z_k^{(s)}$, and fed to the LSTM cell at every step of the text-VAE's decoder.

4 Evaluation

We collected a dataset of lyrics by seven Rock artists: David Bowie, Depeche Mode, Nine Inch Nails, Neil Young, Pearl Jam, Rush, and Doors. Each of them has a distinct musical and lyrical style, and a large catalogue of songs, spanning many years. We intentionally selected artists from the sub-genres of Rock as this models a real-world

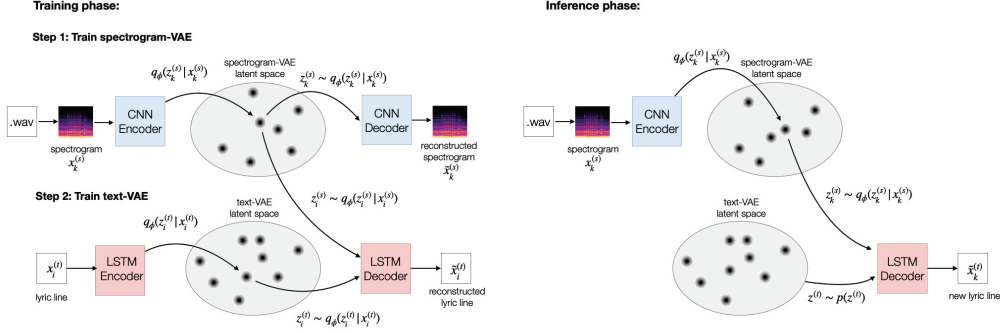


Figure 1: Music audio clip conditioned lyrics generation.

scenario, when a given songwriter might use the model to get influences from the genre congruent with their own work.

Since we do not have access to aligned data for these artists, we manually created a high-precision aligned dataset for two artists (Depeche Mode and Nine Inch Nails, 239 songs), and did an automatic coarse-grained alignment for the other five (518 songs). To create a manually aligned dataset, we annotated the original waveform files in Sonic Visualizer (Cannam et al., 2006) by marking the time corresponding to the start of each lyrics line in the song. The automatic alignment process consisted of splitting each song into 10-second segments and splitting lyrics lines into the same number of chunks, assigning each to the corresponding 10-second clip. In total, the training dataset consists of 18,210 lyrics lines and 14,670 spectrograms.

The goal of the developed model is to generate lyrics lines for an instrumental music piece. Our test set, therefore, only contains instrumental songs: 36 songs from an instrumental album "Ghosts I-IV" by Nine Inch Nails¹ and eight instrumental songs from three other albums by two artists (Depeche Mode and Nine Inch Nails). Each song was split into 10-second clips, which were then converted into spectrograms (807 in total).

First we evaluate the quality of the latent space learned by the spectrogram-VAE. For every spectrogram in the test set, we computed pairwise cosine similarity between its embedding $z^{(s)}$ and the embedding of every spectrogram in the training and test set. We then calculated the proportion of clips in the top 50 and 100 that (a) are part of the

same song, (b) are part of the same album, and (c) belong to the same artist. The results (Table 1) indicate that large proportions of clips most similar to a given clip are by the same artist and from the same album. This demonstrates that spectrogram-VAE learns representations of an artist’s unique musical style.

Top-n	same song	same album	same artist
n=50	0.1707	0.4998	0.7293
n=100	0.0988	0.4462	0.7067

Table 1: Clustering effect in the spectrogram-VAE latent space.

We divided songs in the test set into two categories: “intense” and “calm”. The peak dB differences between tracks in these two categories are statistically significant (t-test, $p < 0.05$). A spectrogram for each 10-second clip was used to generate 100 lines according to the method described in Section 3. The songs in these two categories evoke different emotions and we expect that the lexicon in these two categories of generated lines will be different, but more similar among songs within the same category.

Automatic evaluation of generated lines conditioned on an instrumental audio clip is difficult, since there is no reference ground-truth line that we can compare the generated line to. For this reason, we cannot use n-gram overlap based measures, such as BLEU. Secondly, style-adherence metrics, such as classification accuracy w.r.t. a certain class, e.g. style, in the dataset are inapplicable, since there is no categorical class variable here.

We calculated KL divergence values for words in each song. KL divergence measures the relative entropy between two probability distributions. It was defined in information theory (Losee, 1990) and was formulated as a word ranking measure

¹Ghosts I-IV. Nine Inch Nails. Produced by: Atticus Ross, Alan Moulder, Trent Reznor. The Null Corporation. 2008. Released under Creative Commons (BY-NC-SA) license.

in (Carpineto et al., 2001). Given word w in the generated corpus G_k conditioned on clip k and generated corpus N conditioned on all other clips in the test set, KL divergence is calculated as $\text{word-KL}(w) = p_{G_k}(w) \cdot \log(p_{G_k}(w)/p_N(w))$.

We then calculated rank-biased overlap scores (RBO) to measure pairwise similarity of word-KL ranked lists corresponding to each pair of songs. RBO (Webber et al., 2010) is a metric developed in Information Retrieval for evaluating ranked search results overlap, and handles non-conjoint ranked lists. The RBO score falls in the range $[0,1]$ where 0 indicates a disjoint list and 1 - identical.

Figure 2 shows that “calm” songs have higher RBO values with other songs in the same category, indicating similar generated word distributions, and low RBO values w.r.t. “intense” songs. RBO values for lines generated conditioned on “intense” songs are not as high, suggesting that they have less word overlap. This is likely because there are more songs in this category in the training set with widely different lyrics, therefore the model may be picking up more subtle musical differences, which make it correspondingly generate lyrics that have lyrical influences from different songs.

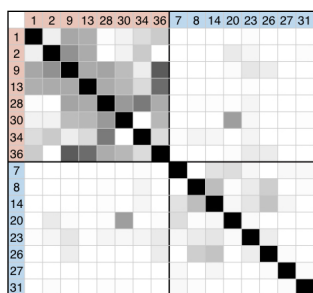


Figure 2: Rank-biased overlap (RBO) between the KL-divergence ranked lists of generated words for songs in “Ghosts I-IV” album. Pink-highlighted songs are calm, and blue - intense. The darker grey cells indicate higher overlap. The row/column headings correspond to the numbers in the song titles (e.g. 2 is for “2 Ghosts I”).

The difference between word distributions is also evident at the audio clip level. Figure 3 shows RBO values for each 10-second clip in the song “12 Ghosts II”². The x-axis is the timeline. We first calculated word-KL for every clip w.r.t. all other clips in the test set. Then pairwise RBO was computed between the given clip’s ranked word list and the ranked word lists for “intense”, and

²Demos of generated lines are available at: <https://sites.google.com/view/nlp4musa-submission/home>

“calm” generated corpora, respectively. The original song’s waveform is given for reference, showing correlation with the change in the lexicon being generated for calm and intense sections of the track.

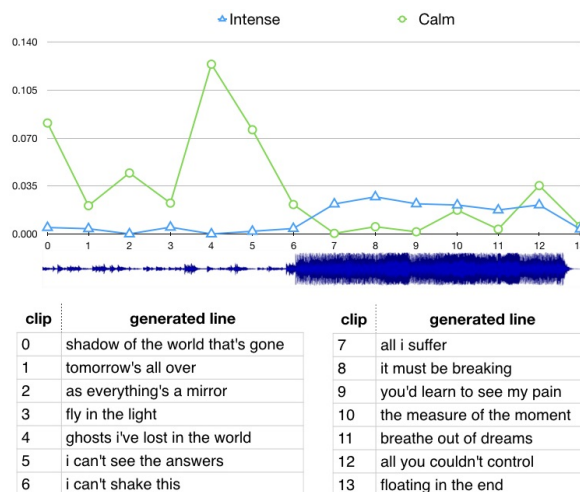


Figure 3: Rank-biased overlap (RBO) values for 10-second clips of “12 Ghosts II” song and examples of lines generated for each clip.

We have also conducted a human evaluation. Six participants (3 female and 3 male), none of whom are members of the research team, were asked to listen to ten instrumental music clips from the test set. For each clip they were given two lists of 100 generated lines. One list was generated conditioned on the given clip in either “calm” or “intense” category, the other list was generated based on a clip from the opposite category. The participants were asked to select the list that they thought was generated based on the given clip. The average accuracy was 78.3% (sd=9.8), which shows that participants were able to detect emotional and semantic congruence between lines and a piece of instrumental music.

5 Conclusions

We developed a bimodal neural network model, which generates lyrics lines conditioned on an instrumental audio clip. The evaluation shows that the model generates different lines for audio clips from “calm” songs compared to “intense” songs. Also, songs in the “calm” category are lexically more similar to each other than to the songs in the “intense” category. A human evaluation shows that the model learned meaningful associations between the semantics of lyrics and the musical characteristics of audio clips captured in spectrograms.

References

- Chris Cannam, Christian Landone, Mark B Sandler, and Juan Pablo Bello. 2006. The sonic visualiser: A visualisation platform for semantic descriptors from musical signals. In *ISMIR*, pages 324–327.
- Claudio Carpineto, Renato De Mori, Giovanni Romano, and Brigitte Bigi. 2001. An information-theoretic approach to automatic query expansion. *ACM Transactions on Information Systems (TOIS)*, 19(1):1–27.
- Cheng-Kuan Chen, Zhufeng Pan, Ming-Yu Liu, and Min Sun. 2019. Unsupervised stylish image description generation via domain layer norm. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8151–8158.
- Wen-Feng Cheng, Chao-Chung Wu, Ruihua Song, Jianlong Fu, Xing Xie, and Jian-Yun Nie. 2018. Image inspired poetry generation in xiaoice. *arXiv preprint arXiv:1808.03090*.
- Diederik P Kingma and Max Welling. 2014. Auto-encoding variational Bayes. In *Proceedings of the International Conference on Learning Representations*.
- Jiwei Li, Michel Galley, Chris Brockett, Georgios Spathourakis, Jianfeng Gao, and Bill Dolan. 2016. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003.
- Yuan Li, Chunyuan Li, Yizhe Zhang, Xiujun Li, Guoqing Zheng, Lawrence Carin, and Jianfeng Gao. 2020. Complementary auxiliary classifiers for label-conditional text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8303–8310.
- Robert M Losee. 1990. *The science of information: Measurement and applications*. Academic Press New York.
- Wenxian Shi, Hao Zhou, Ning Miao, and Lei Li. 2020. Dispersed em-vaes for interpretable text generation. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*.
- Alexey Tikhonov and Ivan P Yamshchikov. 2018. Guess who? multilingual approach for the automated generation of author-stylized poetry. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 787–794.
- Olga Vechtomova, Hareesh Bahuleyan, Amirpasha Ghabussi, and Vineet John. 2018. Generating lyrics with variational autoencoder and multi-modal artist embeddings. *arXiv preprint arXiv:1812.08318*.
- Wenlin Wang, Zhe Gan, Hongteng Xu, Ruiyi Zhang, Guoyin Wang, Dinghan Shen, Changyou Chen, and Lawrence Carin. 2019. Topic-guided variational auto-encoder for text generation. In *HLT-NAACL*, pages 166–177.
- Kento Watanabe, Yuichiroh Matsubayashi, Satoru Fukayama, Masataka Goto, Kentaro Inui, and Tomoyasu Nakano. 2018. A melody-conditioned lyrics language model. In *HLT-NAACL*, pages 163–172.
- William Webber, Alistair Moffat, and Justin Zobel. 2010. A similarity measure for indefinite rankings. *ACM Transactions on Information Systems (TOIS)*, 28(4):1–38.
- Xiaopeng Yang, Xiaowen Lin, Shunda Suo, and Ming Li. 2018. Generating thematic chinese poetry using conditional variational autoencoders with hybrid decoders. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4539–4545.
- Yi Yu, Suhua Tang, Francisco Raposo, and Lei Chen. 2019. Deep cross-modal correlation learning for audio and lyrics in music retrieval. *ACM Trans. Multimedia Comput. Commun. Appl.*, 15(1).
- 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.

Hyperbolic Embeddings for Music Taxonomy

Maria Astefanoaei*
IT University of Copenhagen
msia@itu.dk

Nicolas Collignon*
University of Edinburgh
nccollignon@gmail.com

Abstract

Musical genres are inherently ambiguous and difficult to define. Even more so is the task of establishing how genres relate to one another. Yet, genre is perhaps the most common and effective way of describing musical experience. The number of possible genre classifications (e.g. Spotify has over 4000 genre tags, LastFM over 500,000 tags) has made the idea of manually creating music taxonomies obsolete. We propose to use hyperbolic embeddings to learn a general music genre taxonomy by inferring continuous hierarchies directly from the co-occurrence of music genres from a large dataset. We evaluate our learned taxonomy against human expert taxonomies and folksonomies. Our results show that hyperbolic embeddings significantly outperform their Euclidean counterparts (Word2Vec), and also capture hierarchical structure better than various centrality measures in graphs.

1 Introduction

Music genre is the most popular way to describe music - whether in the context of describing one's listening preferences, or when organising music libraries for efficient user access, exploration and discovery of new music. As is often the case with human categories, genres are richly textured and can be difficult to define explicitly. The rules defining them are ambiguous, complex and dependent on historical, genealogical or geographical factors (Aucouturier and Pachet, 2003). Representing the genre space is therefore particularly challenging. Genre taxonomies are important structures that allow us to represent the relationship between different forms of music. Knowing that *bebop* is a sub-genre of *jazz* that originated from *swing*, and later led to the development of *hard bop* can help better understand and explore the relationships between artists and their music.

The two authors contributed equally.

For long, musicologists held the responsibility of labelling and organising genres. With the digitalisation of music and the rise of internet music consumption, online communities have shown impressive crowd-sourcing efforts in labelling and organising music by sharing structured music knowledge (e.g. DBpedia, LastFM). Beyond the practical use of a structured organisation of music, musical genres also carry an intuitive psychological reality - studies have shown it can take only a quarter of a second for a person to identify the genre of a particular track (Gjerdigen and Perrott, 2008). However, reaching an agreement on what should be considered a genre and how genres are organised amongst each other remains a difficult task. With the ever growing number of music genre labels (e.g. Spotify has over 4000 genre tags, LastFM over 500,000 tags), the effort of manually defining a complete music taxonomy is daunting, and makes capturing the full spectrum of rich interactions between genres beyond reach (Sordo et al., 2008; Pachet et al., 2000).

Historically, the most common approach to represent a music genre space has been by defining trees that capture the hierarchical structure present in genre data sets (see Figure 1).

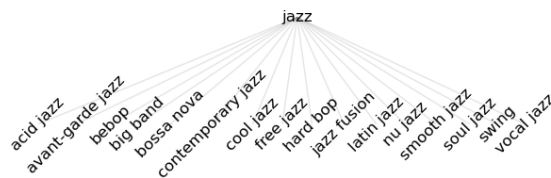


Figure 1: Tree for a subset of the subgenres of jazz.

For example, Schreiber (2015) exploited the asymmetry between main genres and sub genres to construct hierarchies, e.g. alternative is repeatedly associated with rock, whereas rock is associated with a large number of other genres. This asymmetry puts forward rock as a more general genre. Epure et al. (2019) made direct use of the DBpe-

dia music genre ontology to create a translation of music labels across taxonomies.

Embeddings are an alternative approach to capture semantic relatedness by constructing a continuous similarity space. Both approaches face shortcomings: tree-taxonomies fail to address the complexity of the genre space as genres are restricted to one category and measures of similarity between genres of different families are lost. While Euclidean embeddings allow for continuous similarity judgements between music genres, they are unable to capture latent hierarchical information. The goal of this paper is to construct a music genre taxonomy using embeddings in hyperbolic space, which offer a way of representing concepts in a continuous space while preserving hierarchy. We infer this general music genre taxonomy directly from the co-occurrence patterns of music genres across a wide and representative sample of artists.

In the first part of this paper, we introduce hyperbolic embeddings as a powerful method for inferring continuous concept hierarchies. We detail how we created the dataset of genre co-occurrence, our evaluation methods and results.

2 Embedding methods and hyperbolic space

Continuous word representations (Sordo et al., 2008; Levy and Sandler, 2007) are a widely used method to organise words/concepts, and carry many benefits for natural language processing tasks. Typically, the objective of embedding methods is to organize symbolic objects so their similarity in the embedding space reflects their semantic or functional similarity. Word embeddings such as Word2Vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014) and Fasttext (Bojanowski et al., 2017) are widely used for tasks ranging from machine translation to sentiment analysis (Nickel and Kiela, 2017).

Hierarchical structures are a core feature of knowledge representations in many domains. They are used in biology to categorise and represent the relationship between animals of different species, in sociology to understand the organisational structures of groups and communities, in linguistics to describe the origin of languages etc. (Nickel and Kiela, 2018). While explicit hierarchical relations are mostly absent from large datasets, recent developments have focused on inferring this latent hierarchy directly from the data. Nickel and Kiela (2017) introduced the idea of

Poincaré embeddings to learn continuous representations of hierarchies. Their model exploits the geometrical properties of hyperbolic space to capture two aspects of the relationships between embedded items: relatedness and generality; an entity is a parent of another entity if they are related and if the parent is more general than the child. These two aspects can be separated to infer concept hierarchies through hyperbolic embeddings. Relatedness is captured by the distance in the hyperbolic space, while generality is measured through the norm (smaller norm means higher generality). Due to these properties, hyperbolic spaces are particularly suited for embedding discrete trees and graphs with tree-like structure. While the idea of using hyperbolic space to represent hierarchically structured data is not new (see e.g. Lamping et al., 1995; Sarkar, 2011; Kleinberg, 2007), only recently have hyperbolic embeddings attracted the attention of the machine learning community (e.g. Nickel and Kiela, 2017; De Sa et al., 2018; Chamberlain et al., 2017; Liu et al., 2019).

Hyperbolic embeddings have been recently applied to related Music Information Retrieval tasks with success. Schmeier et al. (2019) looked at hyperbolic embeddings for music recommender systems and showed that this led to a significant increase in performance when compared to its Euclidean counterpart. Gunel et al. showed that hyperbolic embeddings could capture the artist and album relationships between tracks. To our knowledge, this paper presents the first attempt at applying hyperbolic embeddings to learn a general music genre taxonomy.

Hyperbolic geometry is a non-Euclidean geometry that emerges from relaxing Euclid’s parallel postulate: for a point not on a line there can pass infinitely many lines parallel to that line. This results in a space that is in some sense larger than the Euclidean counterpart, allowing for low-dimensional embeddings with lower distortion (Sarkar, 2011). A small distortion means that most of the information in the original data is preserved in the embedding. Tree graphs are particularly suited to be represented in hyperbolic space. Sarkar (2011) shows that hyperbolic embeddings of weighted trees can preserve not only the topology of the trees but the tree metric induced by the length of the edges.

There exist multiple, equivalent models of hyperbolic space, such as the Poincaré model and the

Lorentz model. Both have specific strengths: the Poincaré model provides a very intuitive method for visualizing and interpreting hyperbolic embeddings, while the Lorentz model is well-suited for Riemannian optimization, which is particularly useful when considering large datasets. Nickel and Kiela (2018) propose a method that exploits the individual strengths of both these models by building embeddings using the Lorentz model and mapping them into the Poincaré ball. The embeddings are based on large scale, unstructured similarity scores. We use this approach to create hyperbolic embeddings of genres based on their co-occurrence scores, our proposed measure of similarity. In the next section, we explain how we constructed our dataset and the evaluation methods.

3 Creating the dataset

To construct the similarity matrix, we use genre playlists from Every Noise At Once (ENAO)¹ - a Spotify project that attempts to capture the musical genre space. Our dataset consisted of 1368 ENAO playlists, each corresponding to a unique genre, which all tracks in a playlist have in common. Across these playlists, 4309 different genres were represented (see Johnston, 2018, for an idea of how these genres are decided). The ENAO playlists are generated algorithmically and take into account audio features of tracks, user listening patterns and artist relatedness (McDonald, 2013). Spotify attaches genre labels at the artist level, on average 4.9 different genres per artist ($SD = 3.7$). We analyse the artist genre labels within each playlist. The median length of a playlist is 164 tracks ($SD = 181.0$) with at least two different genre tags, and an average of 65.4 different genres ($SD = 54.2$). We compute the similarity matrix by looking at genre co-occurrences across the artists from the tracks in the ENAO genre playlists, following the assumption that these are a fairly accurate representation of the musical genre space. In this context, the list of genres associated to artists can be seen as ‘sentences’ in a more typical word embedding sense. Rather than computing embeddings on the full similarity matrix ($\sim 4000 \times 4000$), we restrain our analysis to the genres matching the ones present in our evaluation datasets.

4 Evaluating Genre Taxonomies

To evaluate our embedding, we compare it to taxonomies defined explicitly by music experts and

¹<http://everynoise.com/>

taxonomies inferred from user annotations (also referred to as *folksonomies*). Four of the five datasets used in the experiments were based on the datasets used in the 2018 AcousticBrainz Genre Task, part of the MediaEval benchmarking initiative (Bogdanov et al., 2017). The dataset in its original form was aimed at testing the automatic genre annotation from content-based features of musical items. These include the All Music dataset, Discogs, LastFM and Tagtraum. The fifth dataset we used was the FMA dataset (Defferrard et al., 2017). Each dataset consists of a set of trees representing different music genres (Figure 1 shows an example from the LastFM dataset).

Dataset	Trees	Genres	Depth	Annotation
FMA	11	68	2	Expert
Allmusic	16	322	2	Expert
Discogs	9	204	1	Expert
LastFM	15	211	1	User
Tagtraum	17	167	1	User

Table 1: Description of the evaluation datasets

We normalise the genre labels across datasets following the rules proposed by Schreiber (2015) and Geleijnse et al. (2007), which included capitalization, spelling standardization, tokenization, and concatenation of the strings. After normalising the genre labels, we find the closest match for each genre from the test set to the genres in the Everynoise playlists. The string matching is based on the Levenshtein distance, which counts the number of edits needed to transform one string to another². We keep genre labels that are above a 0.90 similarity threshold, leaving us with 503 unique genres. Of those, 94% were exact matches.

We use Word2Vec, a popular word embedding method, as baseline (Mikolov et al., 2013). It creates vector representations based on word contexts using shallow neural networks. In our implementation, we consider the genres associated with each track as a sentence, giving us a context for a genre.

5 Results

Using our proposed co-occurrence counts as a pairwise similarity measure between genres, we compute hyperbolic embeddings based on the Lorentz model, following the method proposed by Nickel and Kiela (2018). An example of a 2D embedding mapped to the Poincaré disc is shown in Figure 3. We highlight the genres present in the LastFM trees dataset. Even in 2d, we find the

²We use the FuzzyWuzzy Python package (Cohen, 2011)

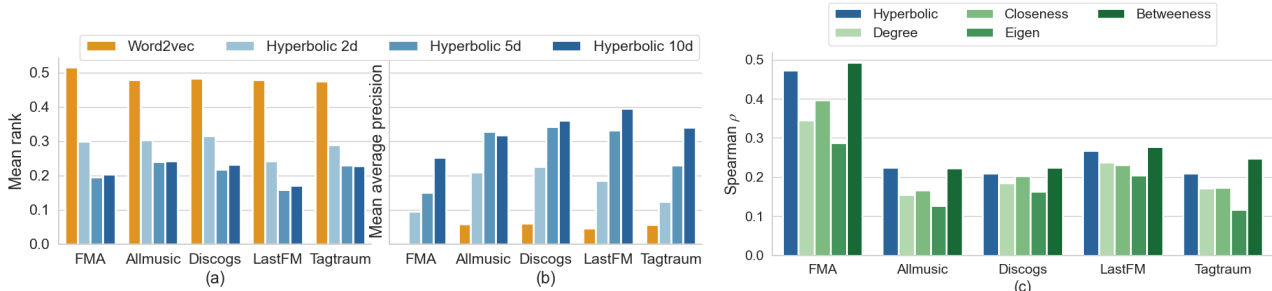


Figure 2: Comparison with Word2vec embeddings and graph centrality measures.

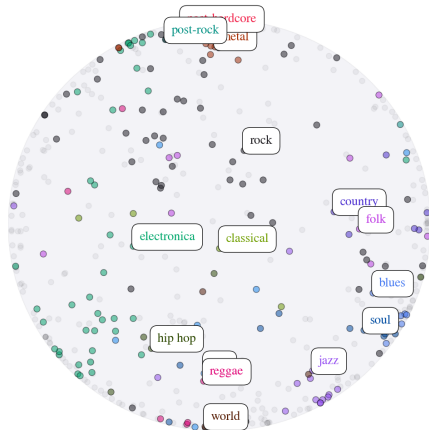


Figure 3: Embedding of the 2d hyperbolic coordinates for genres in the LastFM dataset. All nodes in one tree have the same color. We highlight the the parent genre.

embedding captures the similarity between genres well and highlights meaningful clusters. Parent nodes are typically closer to the center of the disc than their children, suggesting that the embedding successfully represents hierarchical relations. Moreover, we observe natural transitions between clusters of genres, such as *jazz* \rightarrow *soul* \rightarrow *blues* \rightarrow *folk* \rightarrow *country*; these types of transitions cannot easily be captured with disjoint tree representations. Next, we show that hyperbolic embeddings outperform Word2Vec embeddings significantly when evaluated against human expert taxonomies and folksonomies.

To measure the quality of the embedding, for each observed edge (u, v) we compute the corresponding distance $d(u, v)$ in the embedding and rank it among the distances of all unobserved edges for u , i.e., among $\{d(u, v') : (u, v') \notin \mathcal{D}\}$. We then report the mean rank (MR) and mean average precision (MAP) of this ranking. We compare 2D, 5D and 10D hyperbolic embeddings³ to Word2vec (100D)⁴. Hyperbolic embeddings perform better on both metrics regardless of dimension, with the 10d embeddings giving up to 8 fold

³We rely on the following implementation: github.com/theSage21/lorentz-embeddings

⁴We use the *gensim* (Řehůřek and Sojka, 2010) package

precision improvement and 2 fold rank improvement when compared to Word2Vec (Figure 2a,b).

To evaluate how hierarchy is captured by the hyperbolic embedding, we compare the embedding norms with commonly used centrality measures on graph: degree, closeness centrality, eigenvalue centrality and betweenness centrality. Specifically, we evaluate how they correlate to the ranks in the evaluation trees. Across all datasets, we find that the norm of the hyperbolic embeddings is on par with betweenness centrality and consistently outperforms the other centrality measures (Figure 2c). The trees across evaluation datasets are shallow (mostly depth 1, and exceptionally 2) while the embedding gives more granularity of the hierarchy. The shallowness may explain the generally low correlation scores across measures. As further evaluation, we compute the ratio of parent-child relations in trees that were preserved in the embedding (i.e. the parent norm is lower than the child norm). We find that for 2D, 5D, and 10D embeddings, at least 80% of the parent-child were correctly preserved.

6 Conclusion

In this paper, we inferred a general musical genre taxonomy from a large dataset of playlists. We used Lorentz embeddings to learn continuous hierarchies directly from the co-occurrence patterns of genres across tracks. We evaluated our learned taxonomy against human expert taxonomies and folksonomies and found that hyperbolic embeddings significantly outperform their Euclidean counterparts, while also capturing hierarchy better than a number of centrality measures in graphs. Beyond their direct usefulness for computational studies of music, and domains such as music recommendation or genre classification, these results present hyperbolic embeddings as a powerful tool to study other human classification systems, and perhaps for the study of their corresponding psychological representations.

References

- Jean-Julien Aucouturier and Francois Pachet. 2003. Representing musical genre: A state of the art. *Journal of New Music Research*, 32(1):83–93.
- Dmitry Bogdanov, Alastair Porter, Julián Urbano, and Hendrik Schreiber. 2017. The mediaeval 2017 acousticbrainz genre task: Content-based music genre recognition from multiple sources. *CEUR Workshop Proceedings*.
- Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.
- Benjamin Paul Chamberlain, James Clough, and Marc Peter Deisenroth. 2017. Neural embeddings of graphs in hyperbolic space. *arXiv preprint arXiv:1705.10359*.
- Adam Cohen. 2011. Fuzzywuzzy: Fuzzy string matching in python. *ChairNerd Blog*, 22.
- Christopher De Sa, Albert Gu, Christopher Ré, and Frederic Sala. 2018. Representation tradeoffs for hyperbolic embeddings. *Proceedings of machine learning research*, 80:4460.
- Michaël Defferrard, Kirell Benzi, Pierre Vandergheynst, and Xavier Bresson. 2017. **FMA: A dataset for music analysis**. In *18th International Society for Music Information Retrieval Conference (ISMIR)*.
- Elena V Epure, Anis Khelif, and Romain Hennequin. 2019. Leveraging knowledge bases and parallel annotations for music genre translation. *arXiv preprint arXiv:1907.08698*.
- Gijs Geleijnse, Markus Schedl, and Peter Knees. 2007. The quest for ground truth in musical artist tagging in the social web era. In *ISMIR*, pages 525–530. Citeseer.
- Robert O. Gjerdingen and David H. Perrott. 2008. Scanning the dial: The rapid recognition of music genres. *Journal of New Music Research*, 37:100–93.
- Beliz Gunel, Fred Sala, Albert Gu, and Christopher Ré. **Hyperc: Hyperbolic embeddings for entities**.
- Maura Johnston. 2018. **Trap queen and the data scientist: How a subgenre is born**.
- Robert Kleinberg. 2007. Geographic routing using hyperbolic space. In *IEEE INFOCOM 2007-26th IEEE International Conference on Computer Communications*, pages 1902–1909. IEEE.
- John Lamping, Ramana Rao, and Peter Pirolli. 1995. A focus+ context technique based on hyperbolic geometry for visualizing large hierarchies. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 401–408.
- Mark Levy and Mark Sandler. 2007. A semantic space for music derived from social tags. *Austrian Computer Society*, 1:12.
- Qi Liu, Maximilian Nickel, and Douwe Kiela. 2019. Hyperbolic graph neural networks. In *Advances in Neural Information Processing Systems*, pages 8230–8241.
- Glenn McDonald. 2013. **How we understand music genres**.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.
- Maximilian Nickel and Douwe Kiela. 2018. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. *arXiv preprint arXiv:1806.03417*.
- Maximilian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*, pages 6338–6347.
- François Pachet, Daniel Cazaly, et al. 2000. A taxonomy of musical genres. In *RIAO*, pages 1238–1245.
- Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.
- Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. <http://is.muni.cz/publication/884893/en>.
- Rik Sarkar. 2011. Low distortion delaunay embedding of trees in hyperbolic plane. In *International Symposium on Graph Drawing*, pages 355–366. Springer.
- Timothy Schmeier, Joeseeph Chisari, Sam Garrett, and Brett Vintch. 2019. Music recommendations in hyperbolic space: an application of empirical bayes and hierarchical poincaré embeddings. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 437–441.
- Hendrik Schreiber. 2015. Improving genre annotations for the million song dataset. In *ISMIR*, pages 241–247.
- Mohamed Sordo, Oscar Celma, Martin Blech, and Enric Guaus. 2008. The quest for musical genres: Do the experts and the wisdom of crowds agree? In *ISMIR*, pages 255–260.

Computational Linguistics Metrics for the Evaluation of Two-Part Counterpoint Generated with Neural Machine Translation

Stefano Kalonaris
RIKEN AIP
Japan

Thomas McLachlan
RIKEN AIP
Japan

Anna Aljanaki
University of Tartu
Estonia

Abstract

In this paper, two-part music counterpoint is modelled as a neural machine translation (NMT) task, and the relevance of automatic metrics to human-targeted evaluation is investigated. To this end, we propose a novel metric and conduct a user study comparing it to the automatic scores of a base model known to perform well on language tasks along with different models obtained with hyper-parameter tuning. Insights of this investigation are then speculatively extended to the evaluation of generative music systems in general, which still lacks a standardised procedure and consensus.

1 Introduction

The modelling and generation of contrapuntal music has been tackled using a plethora of approaches, ranging from rule and constraint-based (Ebcioglu, 1988; Tsang and Aitken, 1991) to grammars (Gilbert and Conklin, 2007; Quick and Hudak, 2013), statistical methods such as Hidden Markov Models (Farbood and Schöner, 2001; Allan and Williams, 2004), combinations of the latter with pattern-matching models (Cope, 1992) or templates (Padilla and Conklin, 2018), and neural networks. Among the latter, generative adversarial networks (GAN) (Dong et al., 2018), variational autoencoders (VAE) (Roberts et al., 2017), and convolutional neural networks (CNN) (Huang et al., 2017) have proven successful. Recurrent neural networks (RNN), particularly long short-term memory (LSTM) architectures (Sturm, 2018; Simon and Oore, 2017), and more recent attention-based models are also increasingly applied for the generation of music (Payne, 2019; Huang et al., 2018; Hawthorne et al., 2018); however, language-based models have not been employed as much for modelling contrapuntal music.

In a recent study (Nichols et al., 2021), two-part counterpoint generation was treated as a NMT

task, by considering one part as the source language, and the other as the target language (see Figure 1). We extend the NMT analogy from the formulation of the task to the evaluation of the system’s musical output, and consider standard metrics used in translation. A novel variation of a human-based metric is proposed and compared to automatic metrics via a user study, and inter-annotator agreement is also assessed. This paper’s contribution can be summarised as 1) a novel application of computational linguistics methods for the evaluation of counterpoint generated using NMT and 2) reusable insights in the broader domain of generative music systems.

2 Data

We used the Multitrack Contrapuntal Music Archive¹ (MCMA) as the training corpus, comprising only track-separated contrapuntal pieces, each ranging from two to six tracks. The dataset of source-target musical sentences for training our model(s) was obtained by making all $\binom{k_i}{2}$ combinations of pairs of tracks, where i indexes the works. This yielded 1,418 track pairs, which were then segmented into 17,734 non-overlapping four-bar chunks. No data augmentation was performed. Instead, all pieces were normalised to a key with zero flats/sharps (notably, *C/Amin*). Events in each score segment were encoded similarly to (Nichols et al., 2021) although we did not require strictly monophonic voices², and we relied on the model’s inbuilt positional encoding (see Section 3), thus omitting a *beat position* token.

3 Model & Tuning

In this work we have used the *Transformer* model (Vaswani et al., 2017) which allows each

¹<https://mcmareadthedocs.io/en/latest/contents.html>

²splits were encoded as *Chord* objects.



Figure 1: First four bars of the top two voices in J.S. Bach’s Fugue in C minor, BWV 871, from *The Well-Tempered Clavier*, viewed as a NMT compositional task.

position in a sequence to attend (Bahdanau et al., 2015) to any other position. An Encoder or Decoder layer has *Self-Attention*, to attend to any position on the layer input, and the Decoder layer has an additional *Encoder-Decoder Attention* in which each decoding step can attend to all the Encoder final layer outputs. For the finer details we refer the reader to the original paper, aforementioned. There, the authors had success in translation with $d = 512$, $l = 6$, $n_h = 8$, which we refer to as the *Base* model (where d is the dimension of each layer and embedding, l the number of layers, and n_h is the number of attention heads).

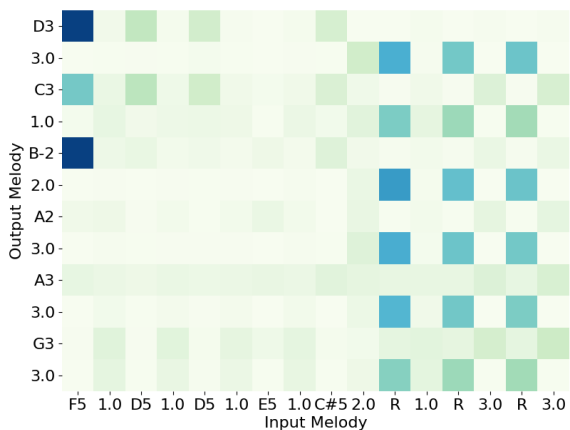


Figure 2: Example of an attention map to the encoder outputs while decoding a short validation phrase. The output is read from top to bottom. (Layer 3, $head_1$)

However, in music generation there are several differences from the language setting, such as having a shared vocabulary between input and output; a reduced vocabulary of 190 tokens compared to the ten thousands to millions of words in language; and considerable differences in the expectation of grammar. In language translation, *attention scores* to the input positions during decoding normally reveal a strong attention along the diagonal. In our music implementation, this is less apparent; however in Figure 2 we see the model attend to positions of rest in the source melody when considering note durations, and to the first few source

pitches when beginning generation of a melody.

These differences motivated the re-tuning of some of the main parameters of the Transformer, namely $l = [2, 4, 6, 8]$, $d = [256, 512, 1024]$, and $n_h = [4, 8, 16]$. In addition, since it was not clear what criteria should be used to halt the training, each model was trained for a set number of epochs ($n_e = [6, 8, 10, 12, 20]$). These epoch ranges were determined by observing the over-fitting behaviour of the base model (up to 60 epochs). This formed 180 parameter sets, which were trained simultaneously on 180 NVIDIA V100 GPUs on *RAIDEN* at *RIKEN* within 24 hours. The question still remained on how these trained models should be evaluated, and how a best model could be selected.

4 Evaluation

In an effort to balance the need for formative assessments aimed at establishing reliable objective measures (Yang and Lerch, 2018) with the necessity to put generative music output back to the domain experts realm (Sturm and Ben-Tal, 2017), we investigated both automatic and human-targeted metrics.

4.1 Automatic metrics

Of the common automatic metrics for translation tasks, we used Loss, Token Accuracy, Bilingual Evaluation Understudy (BLEU) (Papineni et al., 2002), Recall-Oriented Understudy for Gisting Evaluation (ROUGE) (Lin, 2004), Perplexity (Brown et al., 1992), and Word Error Rate (WER) (Klakow and Peters, 2002).

Table 1 shows a few selected candidates favoured by 1 or 2 automatic metrics during tuning. Interestingly, with the exception of *BestPPL*, smaller models were favoured ($d = 256$). The generation quality seemed to be comparable across the selected models, given the amount of data we trained on. So these metrics perform at a similar level when applied to music.

Model	d	l	n_h	n_e	Loss	Acc	WER	BLEU	ROUGE	PPL
<i>Base</i>	512	6	8	10	1.057	0.681	48.16	51.31	74.77	2.904
<i>AccBLEU</i>	256	4	4	20	1.040	0.689	43.53	53.95	76.06	2.862
<i>LossROUGE</i>	256	6	16	8(10)	1.028 (1.022)	0.681	63.54	47.43	76.68	2.812
<i>BestWER</i>	256	2	8	12	1.033	0.682	38.27	53.74	74.53	2.833
<i>BestPPL</i>	512	8	8	8	1.022	0.685	57.03	50.99	75.45	2.798

Table 1: Candidate models selected by 1 or 2 metrics on the validation set. *Acc* refers to Token Accuracy, and *ROUGE* refers to the ROUGE-1 F1 score. The numbers in brackets come from the Loss variant.

Model	NC		NLTM	
	KLD	OA	KLD	OA
<i>Base</i>	0.0026	0.9380	0.0266	0.9696
<i>AccBLEU</i>	0.0008	0.9549	0.0107	0.9540
<i>LossROUGE</i>	0.0011	0.9458	0.0166	0.9498
<i>BestWER</i>	0.0011	0.9499	0.0162	0.9517
<i>BestPPL</i>	0.0017	0.9430	0.0104	0.9576

Table 2: Kullback–Leibler Divergence (KLD) and Overlapping Area (OA) between the models’ dataset intra-set PDF and the inter-set PDF. Shown for total notes used (NC) and note length transition matrix (NLTM).

As a baseline, we used Yang and Lerch’s (2018) evaluation method. Their exhaustive cross-validation based on intra and inter-test measurements, and on Kullback–Leibler Divergence (KLD) and Overlap Area (OA) (see Table 2) also failed to single out a best model.

To get a better understanding of how automatic metrics correlate to music generation quality, we considered human evaluation.

4.2 Human-targeted metrics

Rather than relying on Turing-type tests, which have been sufficiently criticised in (Ariza, 2009), we consider instead the *human-targeted translation edit rate* (HTER) (Snover et al., 2006) and propose a variant that can be used in the music domain. In HTER, typically, human annotators generate a new targeted reference by editing the machine generated target (hypothesis) until it has the same meaning as an original reference translation. Subsequently the *translation edit rate* (TER) is calculated between the new targeted reference and the machine hypothesis.

4.2.1 Music and Semantics

A problem with using HTER ‘as is’, resides in the contentious issue of whether music, as opposed to language, is semantic or not. There seems to

be a general consensus toward the latter, despite studies (Koelsch et al., 2004) showing the ability of music excerpts to prime words. Psychoacoustic and socio-cultural specific properties of music might be able to induce emotion or infer meaning (Meyer, 1956), and there is growing interest in musical semantics (Schlenker, 2017) which, in turn, draw from *Gestalt* theory-based approaches to music (Lerdahl and Jackendoff, 1982). However, it remains unclear how would annotators edit the hypothesis melody so as to have the same “meaning” of the reference target melody. Because of this issue on semantics we consider a simple variation on HTER.

4.2.2 HER

We propose a new metric inspired by the HTER, whereby annotators, all domain experts, are not provided with the (original) reference. Instead, they are asked to edit the generated hypothesis directly until it is, in their domain expertise, sufficiently acceptable as a musical complement/response to the source melody (see Figure 3 for an example). Then, a suitable distance metric (we used the WER) between the obtained targeted reference and the generated hypothesis is calculated. We call this metric, simply, human-targeted edit rate (HER).

5 Experiment

We generated musical mini-scores from the test set for the base model and for the models with the highest validation score on the automatic metrics described in Section 4.1 (AccBLEU, LossROUGE, BestPPL, and BestWER models). The test set models’ targets produced 3,289 mini-scores (each between 2 and 6 bars in length, approximately) for each model. Of these, approximately half (the percentage varied depending on the specific model) were filtered out for



Figure 3: Example of a targeted reference obtained from editing the hypothesis.

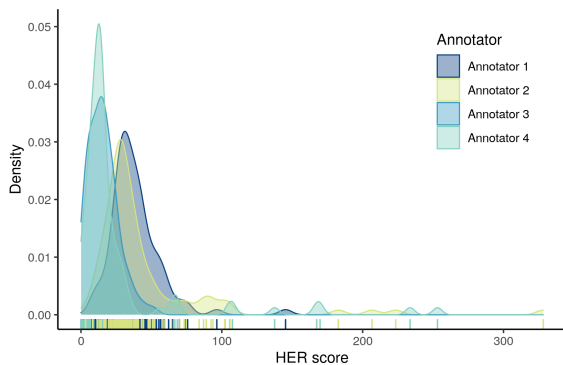


Figure 4: Distributions of the HER scores by annotator displayed as KDE for clarity purposes, as these are easier to visually process than overlapping histograms.

not having an end-of-sequence token, for being badly formatted (not alternating correctly between chord/note/rest and duration tokens) or for having less than three notes in any given part. Subsequently, 1,067 matching mini-scores (across all models) were identified, and 100 of these (20 per model) were randomly selected to be given to 4 annotators. The LossROUGE model scored the lowest mean HER (29.69 ± 21.85). We calculated inter-annotator agreement using the Krippendorff’s alpha coefficient (Krippendorff, 2004) and the intraclass correlation coefficient (ICC) for a fixed set of annotators rating each target (Bartko, 1966). We note that LossROUGE is the least reliable in terms of agreement. The rest of the models range between poor to moderate agreement. The overall inter-model agreement stood at Krippendorff’s alpha of 0.388 and ICC of 0.411. The average amount of edits varied by annotator according to their personal error tolerance, creating a variability in HER. After normalisation, we obtain Krippendorff’s alpha of 0.483 and ICC of 0.61. These results are summarised in Table 3 and in Figure 4.

Model	Mean \pm Std	Kr. α	ICC
<i>Base</i>	31.45 \pm 35.53	0.493	0.431
<i>AccBLEU</i>	35.54 \pm 46.2	0.410	0.452
<i>LossROUGE</i>	29.69\pm21.85	0.164	0.260
<i>BestWER</i>	32.75 \pm 37.20	0.306	0.374
<i>BestPPL</i>	30.54 \pm 25.38	0.493	0.322

Table 3: Intra-model HER scores and agreement.

6 Conclusion & Reusable Insights

We presented a study on computational linguistic metrics applied to the evaluation of two-part music counterpoint generated with a language-based model. A novel human-targeted metric (HER) was proposed, to correlate automatic translation metrics to human judgement, in the music domain. The HER metric bypasses the contentious notion of human/machine discrimination and, while subjectivity is still part of the process (no two annotators would edit the generated hypothesis in an identical way), it does not require defining musical features of interest in advance. It is, instead, assumed that domain practitioners have their own definitions of musical fitness and edit the model’s output accordingly, and that individual biases can be measured via inter-annotator reliability.

In our study, we hoped that the HER score would help elucidate the strength of NLP automatic translation metrics for music generation. While this study proved inconclusive given the low inter-annotator agreement, also reported in other music annotation tasks (Gjerdingen and Perrott, 2008; Flexer and Grill, 2016; Koops et al., 2019), it nevertheless provides an original approach which can be employed to evaluate other generative music systems.

References

- Moray Allan and Christopher K. I. Williams. 2004. Harmonising Chorales by Probabilistic Inference. In *Proceedings of the 17th International Conference on Neural Information Processing Systems, NIPS'04*.
- Christopher Ariza. 2009. [The Interrogator as Critic: The Turing Test and the Evaluation of Generative Music Systems](#). *Computer Music Journal*, 33(2):48–70.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. [Neural Machine Translation by Jointly Learning to Align and Translate](#). In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- John J. Bartko. 1966. The Intraclass Correlation Coefficient as a Measure of Reliability. *Psychological Reports*, 19(1):3–11.
- Peter F. Brown, Vincent J. Della Pietra, Robert L. Mercer, Stephen A. Della Pietra, and Jennifer C. Lai. 1992. An Estimate of an Upper Bound for the Entropy of English. *Computational Linguistics*, 18(1):31–40.
- David Cope. 1992. Computer Modeling of Musical Intelligence in EMI. *Computer Music Journal*, 16(2):69–83.
- Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. 2018. MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*.
- Kemal Ebcioglu. 1988. An Expert System for Harmonizing Four-Part Chorales. *Computer Music Journal*, 12(3):43–51.
- Mary Farbood and Bernd Schöner. 2001. Analysis and Synthesis of Palestrina-Style Counterpoint Using Markov Chains. In *Proceedings of International Computer Music Conference, ICMC'01*.
- Arthur Flexer and Thomas Grill. 2016. [The Problem of Limited Inter-rater Agreement in Modelling Music Similarity](#). *Journal of New Music Research*, 45(3):239–251. PMID: 28190932.
- Édouard Gilbert and Darrell Conklin. 2007. A Probabilistic Context-Free Grammar for Melodic Reduction. In *International Workshop on Artificial Intelligence and Music, The Twentieth International Joint Conference on Artificial Intelligence, IJCAI'07*.
- Robert O. Gjerdingen and David Perrott. 2008. [Scanning the Dial: The Rapid Recognition of Music Genres](#). *Journal of New Music Research*, 37(2):93–100.
- Curtis Hawthorne, Anna Huang, Daphne Ippolito, and Douglas Eck. 2018. Transformer-NADE for Piano Performances. In *NIPS 2018 Workshop on Machine Learning for Creativity and Design*.
- Cheng-Zhi Anna Huang, Tim Cooijmans, Adam Roberts, Aaron Courville, and Douglas Eck. 2017. Counterpoint by Convolution. In *International Society for Music Information Retrieval (ISMIR)*.
- Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, and Douglas Eck. 2018. Music Transformer: Generating Music with Long-Term Structure. *arXiv preprint arXiv:1809.04281*.
- Dietrich Klakow and Jochen Peters. 2002. [Testing the correlation of word error rate and perplexity](#). *Speech Commun.*, 38(1):19–28.
- Stefan Koelsch, Elisabeth Kasper, Daniela Sammler, Katrin Schulze, Thomas Gunter, and Angela D. Friederici. 2004. [Music, language and meaning: brain signatures of semantic processing](#). *Nature Neuroscience*, 7(3):302–307.
- Hendrik Vincent Koops, W. Bas de Haas, John Ashley Burgoyne, Jeroen Bransen, Anna Kent-Muller, and Anja Volk. 2019. [Annotator subjectivity in harmony annotations of popular music](#). *Journal of New Music Research*, 48(3):232–252.
- Klaus Krippendorff. 2004. *Content Analysis: An Introduction to Its Methodology*. Sage.
- Fred Lerdahl and Ray Jackendoff. 1982. *A generative theory of tonal music*. MIT Press, Cambridge, MA.
- Chin-Yew Lin. 2004. [ROUGE: A Package for Automatic Evaluation of Summaries](#). In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.
- Leonard B. Meyer. 1956. *Emotion and meaning in music*. University of Chicago Press, Chicago.
- Eric P. Nichols, Stefano Kalonaris, Gianluca Micchi, and Anna Aljanaki. 2021. Modeling Baroque Two-Part Counterpoint with Neural Machine Translation. In *Proceedings of the International Computer Music Conference*, Santiago, Chile. International Computer Music Association. Preprint available: <https://arxiv.org/abs/2006.14221>.
- Victor Padilla and Darrell Conklin. 2018. [Generation of Two-Voice Imitative Counterpoint from Statistical Models](#). *International Journal of Interactive Multimedia and Artificial Intelligence*, 5(3):22–32.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. [Bleu: a Method for Automatic Evaluation of Machine Translation](#). In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

- Christine Payne. 2019. MuseNet. <https://openai.com/blog/musenet>.
- Donya Quick and Paul Hudak. 2013. Grammar-based Automated Music Composition in Haskell. In *Proceedings of the First ACM SIGPLAN Workshop on Functional Art, Music, Modeling & Design*, FARM'13, pages 59–70, New York, NY, USA. ACM.
- Adam Roberts, Jesse Engel, and Douglas Eck. 2017. Hierarchical Variational Autoencoders for Music. In *NIPS 2017 Workshop on Machine Learning for Creativity and Design*.
- Philippe Schlenker. 2017. Outline of Music Semantics. *Music Perception*, 35(1):3–37.
- Ian Simon and Sageev Oore. 2017. Performance RNN: Generating Music with Expressive Timing and Dynamics. <https://magenta.tensorflow.org/performance-rnn>.
- Matthew Snover, Bonnie Dorr, Richard Schwartz, Lina Micciulla, and Ralph Weischedel. 2006. A Study of Translation Error Rate with Targeted Human Annotation. In *Proceedings of the Association for Machine Translation in the Americas (AMTA)*.
- Bob Sturm. 2018. What do these 5,599,881 parameters mean? : An analysis of a specific LSTM music transcription model, starting with the 70,281 parameters of its softmax layer. In *Proceedings of the 6th International Workshop on Musical Metacreation (MUME 2018)* :. QC 20181106.
- Bob Sturm and Oded Ben-Tal. 2017. Taking the Models back to Music Practice: Evaluating Generative Transcription Models built using Deep Learning. *Journal of Creative Music Systems*, 2(1).
- Chi Ping Tsang and M. Aitken. 1991. Harmonizing Music as a Discipline in Constraint Logic Programming. In *Proceedings of the International Computer Music Conference*, ICMC'91, pages 61–64.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.
- Li-Chia Yang and Alexander Lerch. 2018. On the evaluation of generative models in music. *Neural Computing and Applications*, 32(9):4773–4784.

Interacting with GPT-2 to Generate Controlled and Believable Musical Sequences in ABC Notation

Cariña Geerlings

Department of Computer Science
Vrije Universiteit Amsterdam
The Netherlands
c.geerlings@student.vu.nl

Albert Meroño-Peñuela

Department of Computer Science
Vrije Universiteit Amsterdam
The Netherlands
albert.merono@vu.nl

Abstract

Generating symbolic music with language models is a promising research area, with potential applications in automated music composition. Recent work shows that Transformer architectures can learn to generate compelling four-instrument scores from large MIDI datasets. In this paper, we re-train the small (117M) GPT-2 model with a large dataset in ABC notation, and generate samples of single-instrument folk music. Our BLEU and ROUGE based quantitative, and survey based qualitative, evaluations suggest that ABC notation is learned with syntactical and semantic correctness, and that samples contain robust and believable n-grams.

1 Introduction

Recent advances in deep learning have greatly improved the performance of neural generative systems at automatic music generation. For example, Magenta’s MusicVAE (Roberts et al., 2018) uses hierarchical autoencoders to interpolate novel music samples between different points in a MIDI latent representation. Similar techniques have been proposed for the task of learning language models, mostly in Natural Language Processing (NLP). For example, the Transformer-based neural architectures of BERT (Devlin et al., 2019), GPT-2 (Radford et al., 2019), and Transformer XL (Dai et al., 2019) use encoders/decoders and various attention mechanisms to achieve great performance at language learning and generation. Therefore, it is no surprise that these models have been applied for learning and generating symbolic music scores, assuming that similar sequence-to-sequence attention mechanisms to those of written natural language hold for written music. For example, LakhNES (Donahue et al., 2019) and MuseNet (Payne, 2019) use these language models over MIDI music representations, successfully

```
X:1
T:The Legacy Jig
M:6/8
L:1/8
R:jig
K:G
GFG BAB | gfg gab | GFG BAB | d2A AFD |
GFG BAB | gfg gab | age edB |1 dBA AFD :|2 dBA ABd | :
efe edB | dBA ABd | efe edB | gdB ABd |
efe edB | d2d def | gfe edB |1 dBA ABd :|2 dBA AFD |]
```

Listing 1: An example tune in ABC notation.

addressing large scale, multi-instrument, and long sequence MIDI score learning and generation.

However, a shortcoming of these works is that they learn exclusively over MIDI representations, leaving unanswered questions for other genera and datasets. For example, folk and traditional music are typically encoded using ABC notation (Walshaw, 2011). Moreover, such experiments are almost exclusively evaluated using perplexity (Brown et al., 1992) instead of other language evaluation metrics such as BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004). In this paper, we propose to address these issues by adapting the pre-trained small (117M parameters) language model of GPT-2 (Radford et al., 2019) to learn representations of an ABC notation dataset. ABC notation is an ASCII based character set code that facilitates the sharing of music online (see Listing 1). The first lines indicate the tune index in the file (X:); title (T:); time signature (M:); default note length (L:); type of tune (R:); and key (K:). Following this is the tune, with the | symbol separating measures. Notes are displayed with the letters a to g, where lowercase letters and apostrophes denote higher octaves and uppercase letters and commas denote lower octaves. Further punctuation marks represent variations in the tune. We use conditional sampling, feeding the model two measures and letting it generate the sequence remainder. We evaluate these samples quantitatively, using the BLEU and ROUGE metrics in

various n-gram tests for robustness; and qualitative, via a user survey. Our research question is: “To what extent can language models learn robust representations of ABC notation single-instrument folk music?”.

2 Related Work

Many language models derived from results in computer vision have been investigated in recent years, most with successful applications in music learning and generation. For example, long-short term memory (LSTM) (Hochreiter and Schmidhuber, 1997) recurrent models are commonly used for text generating tasks; and hidden Markov models (HMM) (Rabiner and Juang, 1986) have been used for e.g. speech recognition. More recently, advances in encoder/decoder neural architectures have produced so-called Transformer models, like BERT (dev); OpenAI’s GPT-2 (Radford et al., 2019) –a sequence to sequence transformer with an attention mechanism; and Transformer XL (Dai et al., 2019), a high performance transformer with high compute requirements. The application of these models to music generation has produced various results. For example, OpenAI’s Jukebox (Dhariwal et al., 2020) produces high-fidelity music in the raw audio domain. However, we consider here the language models that can be applied to *symbolic* music generation. In this area, MusicVAE (Roberts et al., 2018) uses a hierarchical variational autoencoder to learn an interpolable latent space of MIDI representations. The works closest to ours are MuseNet (Payne, 2019) and LakhNES (Donahue et al., 2019); in these, authors re-train a Transformer model pre-trained on the Lakh MIDI dataset (Raffel, 2016), a large collection of 176,581 unique MIDI files, to generate four-instrument scores. Our approach is inspired by these works, but focuses on: (a) using GPT-2 instead of Transformer XL, due to the former’s excellent text generation capabilities and left-to-right training; and (b) learning ABC representations of folk and traditional music, rather than using cross-domain MIDI files.

3 Methodology

First, the original data set ¹ was cleaned and all samples were put into separate files. This data set was then used to fine-tune the GPT-2 model on.

¹See <https://www.gwern.net/GPT-2-music>

GPT-2 is a large language model based on the Transformer architecture (Vaswani et al., 2017) with 1.5 billion parameters, trained on a dataset of 8 million web pages with the goal of predicting “the next word, given all of the previous words within some text” (Radford et al., 2019). This model performs very well in a variety of different NLP tasks, and can be re-trained using other datasets and used for generating conditional synthetic text samples. Here, we use retraining on ABC notation —instead of English texts—, and consequently predict the next ABC token that most probably follows all the previous ABC tokens, according to the training data.

During the training phase GPT-2 develops an understanding of the context of the melodies. The fine-tuning is done with all parameters set to default and is stopped, when the loss barely decreases over a large amount of time. This final model will be used to create conditional samples by feeding the model a short musical sequence of two measures from an existing song and letting it generate a subsequent sequence. From the output, another two measures are taken. The two measures from the original song and the generated part are combined to form the new input sequence. This process is repeated, alternating measures from the original song with measures that are generated by GPT-2. Then, these samples are evaluated on their syntax and semantics and they are evaluated using BLEU, ROUGE and a user evaluation form.

BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004) are often utilized in the text processing field to measure the similarity between a machine translated sentence and a human translation. This is based on the number of overlapping n-grams. Similarly, melodies often consist of recurring patterns of consecutive notes. Here, we propose to use these metrics to measure the similarity between a machine generated sequence of ABC notes —through the previous GPT-2 re-training process—, and human-made ABC notations that occur in similar contexts. However, since music is very subjective as well, a user evaluation is used in addition. The outcome of these evaluations will determine whether valid, but also fluent musical pieces can be generated, by having some control over the process.

4 Experiment

The 117M parameters model was used for this, considering the limited amount of time and the fact that larger models might overfit. Furthermore, the longer the model is trained, the better it can familiarize itself with the training data. This often increases the chances of a good performance. This is why the training is stopped when the loss hardly decreases over a substantial amount of time. The model alternated between an average cross-entropy loss of 0.86 and 0.94 over several hours, meaning the model had a hard time optimizing further from this point on. The resulting model was used to generate controlled sequences of music. Two songs from the used data set were chosen and two songs from the left out data set were chosen to diversify. Firstly, the first two measures of an original song are fed, including the header. Based on this, the model is then prompted to generate notes that follow the sequence. From the outcome, only the first two measures are added to the input. The resulting, larger sequence will be fed to the model again, so it can extend this sequence with two measures as well. This is repeated three times, to obtain a song of 12 measures, that consists of 6 measures from the original song and 6 measures generated by the model, alternately.

4.1 Quantitative Evaluation

The similarity between the original melodies and the samples are calculated using the BLEU and ROUGE metrics. Two tables are displayed for the n-grams of BLEU and ROUGE scores for each sample.

BLEU scores				
	1-gram	2-gram	3-gram	4-gram
Sample 1	0.60	0.51	0.48	0.46
Sample 2	0.71	0.57	0.48	0.45
Sample 3	0.56	0.47	0.44	0.42
Sample 4	0.76	0.60	0.54	0.52

Table 1: The BLEU scores for all samples over n-grams 1 to 4

The BLEU score measures how many bi-grams from the GPT-2 generated samples occur in the original song. The scores can range from 0 to 1. 0 indicating no overlap with the original song, 1 indicating a perfect overlap with the original song. Since, half of a sample is copied from the original song, the precision should not go much be-

ROUGE scores		
	1-gram	2-gram
Sample 1	0.62	0.53
Sample 2	0.72	0.58
Sample 3	0.89	0.74
Sample 4	0.77	0.60

Table 2: The ROUGE scores for all samples over n-grams 1 to 4

low 0.50. However, this might occur, when the generated sample has less tokens than the original song, which is the case in sample 3. Samples 1 and 2 have some, but not excessive overlap with their originals. While the fourth sample has many overlapping bi-grams with the original song. The ROUGE score computes the number of bi-grams from the original song that occur in the generated sample. Samples 1, 2 and 4 overlap a little more than 50%, keeping in mind that this might be caused by the length of the sample. Sample 3 shows that numerous bi-grams overlap with the generated sample.

4.2 Qualitative Evaluation

The questionnaire yielded 83 responses. Roughly half of these were male and half were female, with one person preferring not to specify this. Slightly more than 50% of the participants were between the age of 10 and 25, while the rest was older. Most candidates were educated on the level of a Bachelor’s degree. About a quarter is educated higher than this and the remaining quarter is educated lower or not at all. 52% of participants were students, of which 12% had either a full-time or part-time job as well. Another 41% was occupied by solely a full-time job, while the remaining percentage either had a part-time job, was unem-

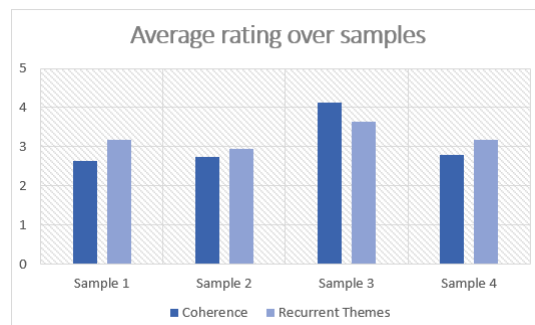


Figure 1: The average ratings of the questionnaire by sample

```

X: 129531
M: 6/8
K: Cmaj
^c2^A^G^G^G|^c^A^A^A2^G|

K: Cmaj
|: CDECDE | =F2GA2G |

|^c2^A^G^G^G|^c^A^A^A2^G|

M: 3/4
K: Cmaj
|: =C=B, =C=F=G, =C| =B, =D=D=F2=G |

|=f2^g=f^c^d|=f^c^A^A2^G|

=A=G=E=c2|1=E=C=B, =D=C|

```

Listing 2: The first sample of GPT-2's generated ABC notation.

played or had another occupation. As expected over half of the participants were Dutch. The other nationalities are spread over 15 other countries. As for the musical knowledge, half of the participants scored themselves below average, approximately 20% thought they were (close to) an expert and over a quarter thought they had an average level of musical knowledge.

Regarding the scoring of the samples, the questions were answered by a rating from 1 to 5. Two existing songs were used as a baseline, of which the average scores were 3.7 and 3.9 for coherence and 2.9 and 3.5 for recurrence. The first sample got an average scoring of 2.6 for coherence and 3.1 for the amount of recurrence. The second sample got a coherence of 2.7 and was scored 2.9 for recurrence. The third sample had a coherence of 4.1 and a recurrence of 3.6. The fourth sample had a coherence of 2.6 along with a scoring of 2.5 for recurrent themes. The two samples that contained existing songs got a score of 3.7 and 3.9 for coherence and a score of 2.9 and 3.5 for recurrence.²

4.3 Syntax and semantics

The first and second samples are presented, where the areas in bold are generated by GPT-2. The third and fourth samples can be found on Dropbox.³ When looking at the meter of the first sample, which is 6/8, the model mostly adheres to it, until it changes the meter to 3/4. After this, the model still holds on to the first meter and in the last generated part follows neither. Furthermore, the model specifies what key and meter it is using,

²See <https://soundcloud.com/user-512999768>

³See https://www.dropbox.com/s/orjvc2mx0sirtti/melody_samples.pdf?dl=0

```

X: 129557
M: 12/8
K: Cmaj
|^C2=F^G2^G|^A^c^A^G=F^D|

=F=E/2=D/2=C=F=G=A |=G=F=D=F2=A, |

^C2=F^G2^G|^A^c^A^G=F^G|

L: 1/8
K: Gmaj
|: D2G2GF | DEGABC |

=f^d^c=c^A^G|^A^c^A^G=F^G|

M: 6/8
K: Cmaj
|: ^C^D=F^C^G^F|^G^C^c^G^F^A|

```

Listing 3: The second sample of GPT-2's generated ABC notation.

even though this key is the same as the given key. What stands out is that the model barely uses the caret, in spite of its high frequency in the original song. On top of this, the model seems to have a tendency to use equality signs, which represents an unaltered pitch of a note. The melody of sample 2 is syntactically flawed. A colon is used to open a repetition, however it is never closed. This happens in the second and third generated parts. The meter is 12/8 in the beginning and changed to 6/8 in the last generation. The key is changed in the last two generations, first to G major and then back to C major. Another noticeable concept is that in the first generation the notes are all naturalized, while this is uncommon in the original song. However, the carets, that are frequent, are not adopted until the last generation.

5 Conclusion

Influencing the generation process of samples led to reasonable results. The model does not deviate far from correct syntax and semantics. Furthermore, plausible results are obtained using the BLEU and ROUGE metrics. This can be deducted from the small decrease in performance while the n-grams increase. The user evaluation showed around average or higher ratings for each of the samples obtained from users with different backgrounds. These results are reason to believe that this method can result in robust musical sequences. However, an improvement may be to use a larger data set to increase the models performance. Or one might choose to use another language model altogether, such as those mentioned in the related work section. More metrics from the field of NLP can be added to see how this would relate to the BLEU and ROUGE scores.

References

- Peter F Brown, Stephen A Della Pietra, Vincent J Della Pietra, Jennifer C Lai, and Robert L Mercer. 1992. An estimate of an upper bound for the entropy of English. *Computational Linguistics*, 18(1):31–40.
- Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.
- Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. 2020. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*.
- Chris Donahue, Huanru Henry Mao, Yiting Ethan Li, Garrison W Cottrell, and Julian McAuley. 2019. Lakhnes: Improving multi-instrumental music generation with cross-domain pre-training. In *International Society for Music Information Retrieval Conference*, pages 685–692.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.
- Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.
- Christine Payne. 2019. MuseNet. <https://openai.com/blog/musenet/>.
- Lawrence Rabiner and B Juang. 1986. An introduction to hidden Markov models. *iee assp magazine*, 3(1):4–16.
- Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.
- Colin Raffel. 2016. *Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching*. Ph.D. thesis, Columbia University.
- Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. 2018. A hierarchical latent vector model for learning long-term structure in music. *arXiv preprint arXiv:1803.05428*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.
- Chris Walshaw. 2011. The ABC music standard 2.1. Technical report, abcnotation.com.

BUTTER: A Representation Learning Framework for Bi-directional Music-Sentence Retrieval and Generation

Yixiao Zhang Ziyu Wang Dingsu Wang Gus Xia

Music X Lab, NYU Shanghai

{yz6492, zz2417, dw1920, gx219}@nyu.edu

Abstract

We propose BUTTER, a unified multi-modal representation learning model for Bi-directional mUSIC-senTence ReTRetrieval and GenERation. Based on the variational autoencoder framework, our model learns three interrelated latent representations: 1) a latent music representation, which can be used to reconstruct a short piece, 2) keyword embedding of music descriptions, which can be used for caption generation, and 3) a cross-modal representation, which is disentangled into several different attributes of music by aligning the latent music representation and keyword embeddings. By mapping between different latent representations, our model can search/generate music given an input text description, and vice versa. Moreover, the model enables controlled music transfer by partially changing the keywords of corresponding descriptions.¹

1 Introduction

The ability to relate natural language descriptions with music is of great importance. It is useful for cross-modal *music analysis*, such as automatic music captioning and music retrieval based on natural language queries. It also has considerable research value in cross-modal *controlled music generation*, say, automatically compose a piece of music according to text descriptions.

While traditional machine-learning algorithms mostly consider analysis (from data to labels) and controlled generation (from labels to data) two completely different tasks, recent progress in multi-modal representation learning (Baltrušaitis et al., 2018) suggests that the two tasks can be unified into a single framework. Specifically, music and the corresponding text descriptions can be regarded

¹Codes are available at <https://github.com/ldzhangyx/BUTTER>

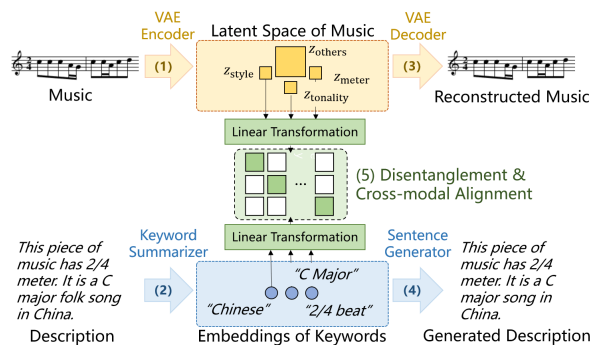


Figure 1: An overview of the model architecture.

as data (of two different modalities) with shared latent representations. Therefore, if we can successfully learn the shared cross-modal representation, music analysis and generation would simply refer to various ways of mapping between such representation and data of different modalities.

Inspired by the idea above, we contribute a multimodal representation learning model for bi-directional music-sentence retrieval and generation. Figure 1 shows the overall model architecture. Here, the yellow path shows music representation learning, the blue path shows keywords summarization and text generation of music description, and the green part represents the cross-modal alignment between the latent music space and keyword embeddings. The cross-modal alignment helps disentangle the latent music representation into four factors: meter, key, style, and others, in which the first three factors have corresponding keywords descriptions. During the inference time, this model enables a number of applications:

- **Task 1: Music retrieval by text description**, where the information flow is (1)→(5)←(2). That is, to search the music segments whose latent representations best correlated with the keywords of the text description in the cross-modal space.
- **Task 2: Music captioning**, where the informa-

tion flow is (1)→(5)→(4). E.g., to output “this is a British style song in C major of 4/4 meter” given a segment of music.

- **Task 3: Controlled music generation**, where the information flow is (2)→(5)→(3). This is very similar to task 1, except that we are now generating/sampling new music rather than searching existing music in the dataset.
- **Task 4: Controlled music refinement (style transfer)**, where the information flow is (2)→(5)→(3)←(1). That is, to learn the latent representation of piece and then refine it by partially changing the keywords of the text description. This task helps us answer the imaginary questions, such as “*what if* a piece is composed in a different style”.

In sum, the contributions of our paper are:

- We designed the first multimodal representation-learning framework which unifies music analysis and controlled music generation.
- We show that music-text alignment in the latent space serves as an effective inductive bias for representation disentanglement. Such disentanglement leads to controllable attributes of music via natural language under weak supervision with no need to do feature engineering for each separated attribute.

2 Related Work

Cross-modal retrieval task has attracted researchers for decades (Gudivada and Raghavan, 1995). Frome et al. (2013) uses a ranking cost to map images and phrases into a common semantic embedding. Yu et al. (2019) uses CCA to model cross-modal relation between audio and lyrics for bi-directional retrieval. Feng et al. (2014) learns multi-modal representations by correlating hidden representations of two uni-modal autoencoders and minimizes a linear combination error. Karpathy et al. (2014) proposed a bidirectional image-sentence mapping method by extracting local fragments. Compared with cross-modal retrieval, cross-modal controlled generation is in general a more difficult task since it requires reconstruct or sample new data from the latent representation. Recent works include automatic image captioning (Xu et al., 2015; Chen and Lawrence Zitnick, 2015; Jia et al., 2015) and text-to-image generation (Hinz et al., 2019; Zhu et al., 2019; El-Nouby et al., 2019). However, very few of them consider the bi-directional generation problem.

Another related area to this study is representation disentanglement. Locatello et al. (2019) shows that the key to a successful disentanglement is to incorporate the model with proper inductive biases. Speaking of the disentanglement for music, Deep Music Analogy (Yang et al., 2019) is very relevant to this study as it can disentangle pitch and rhythm factors. However, the inductive bias comes from a supervision (an explicit rhythm loss), while our study uses text descriptions as a much weaker supervision as well as a more natural form of inductive bias.

3 Method

3.1 Music Modality

We use a similar data representation as in MusicVAE (Roberts et al., 2018). Each 16-beat melody segment x is represented as a sequence of 64 one-hot vectors. Each vector represents a 16th note and has 130 dimensions, representing 128 MIDI pitches, hold and rest, respectively.

We use the VAE framework to learn the latent code z of a melody segment (as shown in (1) and (3) of Figure 1). We assume z conforms to a standard Gaussian prior (denoted by $p(z)$), and can be partitioned into four disentangled factors $z = [z_{\text{key}}, z_{\text{meter}}, z_{\text{style}}, z_{\text{others}}]$, where z_{others} represents the music information not covered by key, meter or style. The VAE encoder uses a single layer bi-directional GRU to encode the melody and emit the mean and variance of the approximated posterior $q_{\theta}(z|x)$. We assume $q_{\theta}(z|x)$ is isotropic Gaussian and denote its mean as $e = [e_{\text{key}}, e_{\text{meter}}, e_{\text{style}}, e_{\text{others}}]$. For the VAE decoder, we apply a 2-layer GRU which outputs $p_{\theta}(x|z)$.

We define the *reconstruction objective* by the ELBO (evidence lower bound) (Kingma and Welling, 2013) as follows,

$$\mathcal{L}_r(\phi, \theta; x) = -\mathbb{E}_{z \sim q_{\phi}} \log p_{\theta}(x|z) + \alpha \text{KL}(q_{\phi} || p(z)), \quad (1)$$

where α is a balance parameter.

3.2 Language Modality

3.2.1 Keywords Representations

We define the keywords of a music description as a triplet $[w_{\text{key}}, w_{\text{meter}}, w_{\text{style}}]$, where $w_{\text{key}} \in D^{\text{key}}$, $w_{\text{meter}} \in D^{\text{meter}}$, and $w_{\text{style}} \in D^{\text{style}}$. Here D^{key} , D^{meter} , and D^{style} are the dictionaries of the three corresponding attributes. We define the overall dictionary $D = D^{\text{key}} \cup D^{\text{meter}} \cup D^{\text{style}}$ and

embed every keyword $w \in D$ to a $|D|$ dimensional one-hot vector e'_w .

3.2.2 Summizer and Generator Module

We apply two GRU-based encoder-decoder models as the keyword summarizer and the description generator, respectively. Both models are pre-trained with sentence-keywords pairs directly retrieved from the dataset. This procedure is shown in (2) and (4) in Figure 1.

3.3 Cross-modal Alignment

We use two linear transformations $f(\cdot)$, $g(\cdot)$ to map latent melody representation and keywords to a shared latent space. We employ a *similarity objective* \mathcal{L}_a to align two representations by maximizing correlation of corresponding attributes while minimizing the correlation of irrelevant attributes. Formally,

$$\mathcal{L}_a = 3 - \sum_{i \in \mathcal{I}} \left(\frac{\langle u_i, v_{w_i} \rangle}{|u_i| \cdot |v_{w_i}|} - \beta \sum_{\substack{w \in D \\ w \neq w_i}} \left| \frac{\langle u_i, v_w \rangle}{|u_i| \cdot |v_w|} \right| \right), \quad (2)$$

where $\mathcal{I} = \{\text{key, meter, style}\}$, $u_i = f(e_i)$, and $v_w = g(e'_w)$. Here, 3 is a constant to keep the loss term non-negative and β is a balance factor. Hence, the overall loss \mathcal{L} is calculated by $\mathcal{L} = \mathcal{L}_r + \gamma \mathcal{L}_a$, where γ is a parameter for balancing two losses.

In theory, the cross-modal alignment module allows bi-directional music-sentence retrieval and generation by inference-time optimization of eq. 2. In practice, we find the keyword combination that best describes a given melody (i.e., minimizes eq. 2) by a brute-force search. Conversely, we compute the latent music code corresponding to a keyword by averaging the latent codes of all music samples aligned with the same keyword.

4 Experiments

We conduct two experiments to demonstrate that the proposed model can be applied to the four tasks mentioned in the introduction.

The former two tasks, i.e., music retrieval by text description and music captioning are both about *music analysis*. The core of the two tasks requires the latent code being able to classify to the correct keywords. Our first experiment (Section 4.2) focuses on this classification accuracy.

The latter two tasks, controlled music generation and controlled music refinement, require that by changing the latent codes (e.g., from minor to

major key), the generated samples also have the corresponding change (e.g., key change) *while still preserving high music quality*. We conduct subject evaluations regarding this aspect in Section 4.3.

4.1 Dataset and Training

Our dataset contains 16,257 folk songs paired with metadata collected from the abc notation homepage². From metadata we select *key*, *meter* and *style* as keywords, and we synthesize diverse description sentences by human craft and paraphrasing tools³. We associate them with 4-bar music segments of corresponding songs. We use 80% for training, 10% for validation and 10% for testing.

We train our model on two keyword settings. In the *full version*, the key keyword contains 25 classes including 24 major/minor keys and *others*; the meter keyword contains 6 classes including *2/4*, *3/4*, *4/4*, *6/8*, *9/8* and *others*. In the *easy version*, the key keyword contains *major*, *minor*, and *others*; the meter keyword contains *triple*, *duple*, and *others*. In both modes, the style keyword contains 3 classes, including *Chinese*, *English*, *Irish*.

We set the size of GRU hidden states, latent variables and attribute variables to 512, 256 and 32 respectively. We map the latent z and words to a shared 32-D embedding space. During training, we set batch size to 4 and learning rate to $1e-3$ with weight decay of 0.999. We set balancing factors $\alpha = 0.01$, $\beta = 0.2$ and $\gamma = 0.1$.

4.2 Objective Measurements for Cross-modal Music Information Retrieval

We design a classification task to evaluate whether the latent codes e_{key} , e_{meter} and e_{style} can predict the corresponding keywords by the similarity objective eq. 2. If so, it follows that with simple algorithms the model is capable of the task *music retrieval by text description* and *music captioning*.

To this end, we compare our models with 2 baseline classifiers under both *full version* and *easy version*. Both baseline models uses the same GRU encoder ((1) in Figure 1) and replace the alignment module ((5) in Figure 1) by three separate MLP classifiers for the three keyword attributes accordingly. Each MLP has 3 linear layers with 128 hidden dimensions. The first baseline method (**GRU-MLP**) trains the whole network from scratch, and the second baseline (**Latent-MLP**) method trains only the MLP classifiers and fixes the GRU encoder

²<http://abcnotation.com/>

³<https://quillbot.com/>

Model	Key		Meter		Style
	Full	Easy	Full	Easy	
GRU-MLP	0.63	0.76	0.44	0.54	0.84
Latent-MLP	0.45	0.83	0.38	0.72	0.90
Ours	0.60	0.77	0.40	0.76	0.92

Table 1: Performance of models in classification task. parameters. Table 1 shows the evaluation results.

When all the keywords of the melody are determined, our model generates complete sentences through the description generator. Figure 2 provides two generated examples.



Melody	
Caption	This is a song in G . It has a 2/4 meter and it is a Chinese song.
Melody	
Caption	This is a 6/8 -meter composition in G major . It is an Irish song.

Figure 2: Generated descriptions for input melodies.

4.3 Subjective Evaluation for Controlled Music Generation

We wish that when we change one or more keyword attributes, the generated music would also make corresponding change while still preserving good musicality. The *controlled music generation* and *controlled music refinement* tasks directly follow from this desired property.

We invite people to subjectively rate the quality of the generated music. In particular, we ask the subjects to listen to 30 samples randomly picked from the test dataset with three types of processing, and each type contains 10 samples:

- (Original)** No processing: identical to the data sample.
- (Ours)** Randomly change the latent code (among e_{key} , e_{meter} and e_{style}) into a target latent code. E.g. to substitute $e_{\text{key}=\text{major}}$ as $e_{\text{key}=\text{minor}}$
- (Prior)** Randomly change the latent code into Gaussian noise sampled from the prior distribution.

The subjects are asked to:

- Rate the musicality** of the processed sample based on a 5-point scale from 1 (low) to 5 (high).
- Select the keyword attributes that best describe the music.** That is, 1) whether the *key* is major, minor or others, 2) whether the *meter* is duple, triple or others, and 3) whether the *style* is Chinese, Irish or English.

Model	Musicality	Human Accuracy		
		Key	Meter	Style
Original	3.44	0.57	0.60	0.38
Prior	2.68	0.33	0.43	0.31
Ours	3.25	0.35	0.48	0.49

Table 2: Performance of models in generation task. Musicality means the overall quality of music.

A total of 30 subjects (9 female and 21 male) participated in the survey. 10% of them are at the professional level of musicological knowledge and the 37% are over the average level. In table 2, the left column shows the rating of musicality and the right column shows the accuracy of selecting the correct keywords. The results show that our proposed method has higher musicality and achieves better control of generation than randomly sampling from the prior in all three factors. However, we still see a gap between our method and the original samples. This is probably because the selected factors deal with deep music structure which remains a challenging task for existing methods, which we leave for future work. Moreover, due to the cultural background of subjects, they generally have difficulty in distinguishing between Irish and English songs. If we combine these two categories into one, then the scores of the original songs, ours, and baseline are: 0.72, 0.59 and 0.32, respectively. Figure 3 shows a transfer example:

Origin (Chinese Style)	
Culture Transfer (to English)	

Figure 3: An example of music refinement.

5 Conclusion and Limitations

In conclusion, we contributed a unified multi-modal representation learning model allowing bi-directional retrieval and generation between music and sentences. The cross-modal alignment serves as an effective inductive bias to disentangle latent representations of music according to text.

We see that the current text description is still very rigid, limited to three keywords and the text descriptions have to cover the exact keywords. In the future, we will make the text description more flexible, covering more music attributes while allowing synonyms. In addition, human descriptions of music may be subjective and ill-defined, making the learning process difficult. It will be the biggest challenge our model faces in the future.

References

- Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2018. Multimodal machine learning: A survey and taxonomy. IEEE transactions on pattern analysis and machine intelligence, 41(2):423–443.
- Xinlei Chen and C Lawrence Zitnick. 2015. Mind’s eye: A recurrent visual representation for image caption generation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2422–2431.
- Alaaeldin El-Nouby, Shikhar Sharma, Hannes Schulz, Devon Hjelm, Layla El Asri, Samira Ebrahimi Kahou, Yoshua Bengio, and Graham W Taylor. 2019. Tell, draw, and repeat: Generating and modifying images based on continual linguistic instruction. In Proceedings of the IEEE International Conference on Computer Vision, pages 10304–10312.
- Fangxiang Feng, Xiaojie Wang, and Ruifan Li. 2014. Cross-modal retrieval with correspondence autoencoder. In Proceedings of the 22nd ACM international conference on Multimedia, pages 7–16.
- Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc’Aurelio Ranzato, and Tomas Mikolov. 2013. Devise: A deep visual-semantic embedding model. In Advances in neural information processing systems, pages 2121–2129.
- Venkat N Gudivada and Vijay V Raghavan. 1995. Content based image retrieval systems. Computer, 28(9):18–22.
- Tobias Hinz, Stefan Heinrich, and Stefan Wermter. 2019. Semantic object accuracy for generative text-to-image synthesis. arXiv preprint arXiv:1910.13321.
- Xu Jia, Efstratios Gavves, Basura Fernando, and Tinne Tuytelaars. 2015. Guiding the long-short term memory model for image caption generation. In Proceedings of the IEEE international conference on computer vision, pages 2407–2415.
- Andrej Karpathy, Armand Joulin, and Li F Fei-Fei. 2014. Deep fragment embeddings for bidirectional image sentence mapping. In Advances in neural information processing systems, pages 1889–1897.
- Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114.
- Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. 2019. Challenging common assumptions in the unsupervised learning of disentangled representations. In international conference on machine learning, pages 4114–4124.
- Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. 2018. A hierarchical latent vector model for learning long-term structure in music. In ICML.
- Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In International conference on machine learning, pages 2048–2057.
- Ruihan Yang, Dingsu Wang, Ziyu Wang, Tianyao Chen, Junyan Jiang, and Gus Xia. 2019. Deep music analogy via latent representation disentanglement. arXiv preprint arXiv:1906.03626.
- Yi Yu, Suhua Tang, Francisco Raposo, and Lei Chen. 2019. Deep cross-modal correlation learning for audio and lyrics in music retrieval. ACM Trans. Multimedia Comput. Commun. Appl., 15(1).
- Minfeng Zhu, Pingbo Pan, Wei Chen, and Yi Yang. 2019. Dm-gan: Dynamic memory generative adversarial networks for text-to-image synthesis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5802–5810.

Unsupervised Melody Segmentation Based on a Nested Pitman-Yor Language Model

Shun Sawada
Future University Hakodate
Hokkaido, Japan
b1012046@gmail.com

Kazuyoshi Yoshii
Kyoto University, Japan
yoshii@kuis.kyoto-u.ac.jp

Keiji Hirata
Future University Hakodate
Hokkaido, Japan
hirata@fun.ac.jp

Abstract

We present unsupervised melody segmentation using a language model based on a non-parametric Bayesian model. We adapt unsupervised word segmentation with a nested Pitman-Yor language model (NPYLM) used in the field of natural language processing to musical note sequences. Treating music as a language, we aim to extract fundamental units, similar to “words” in natural language, from symbolic musical note sequences using a data-driven approach, the NPYLM. We assume musical note sequences generated by the probabilistic model, integrate a note-level n -gram language model and motif-level n -gram language model, and extract fundamental units (motifs) from them. This enables us to conduct melody segmentation, obtaining a language model for the segments, directly from a musical note sequence without annotation. We discuss the characteristics of this model by comparing the rules and grouping structure of a generative theory of tonal music (GTTM).

1 Introduction

In general, a melody is considered to be time series data of notes with various properties such as pitch and duration. We call this time series data a musical note sequence. In the field of musical information retrieval (MIR), the task of melody segmentation, that is, division of a musical note sequence into meaningful units such as motifs and phrases, is one of the most important and fundamental tasks. Melody segmentation, the division of a musical note sequence into meaningful units such as motifs and phrases, is one of the most important and fundamental tasks in the field of musical information retrieval (MIR). Motifs are considered to be one of the most important and fundamental units of music (Lerdahl and Jackendoff, 1996). If we are able to divide a musical note sequence into appropriate motifs, these motifs

can then be used in various tasks such as analyzing a musical structure, automatic composition, and representation learning via “motif embedding” (Hirai and Sawada, 2019).

There are two types of conventional melody segmentation method: rule-based (Lerdahl and Jackendoff, 1996; Cambouropoulos, 2001; Temperley, 2004), and statistic based using properties of musical data (Lattner et al., 2015; Pearce et al., 2010). Although supervised melody segmentation methods have also been proposed (Hamanaka et al., 2017), the cost of producing annotation data of sufficient quality and quantity is enormous. Further, the interpretation of the motifs is subjective and can vary from one annotator to another.

In this study, we aim to extract fundamental units, like “words” in natural language, from symbolic musical note sequences using an unsupervised data-driven approach. It is not known how many notes a motif is made up of, and there are theoretically an infinite number of possible motifs. Therefore, we have to use the vocabulary-free n -gram model instead of the conventional n -gram model, which requires motifs to be defined as a vocabulary in advance. Specifically, we apply an unsupervised word segmentation method, a nested Pitman-Yor language model (NPYLM) (Mochihashi et al., 2009), to a musical note sequence. A sentence in natural language (e.g., English) consists of a combination of words. A word in a natural language consists of a combination of characters. If we think of a character in natural language as equivalent to a musical note and a sentence as equivalent to a note sequence of some length, we can think of the note sequence as consisting of combinations of motifs with units corresponding to words in natural language.

There have been studies that apply the Pitman-Yor language model to music. A hierarchi-

cal Pitman-Yor language model (HPYLM) (Teh, 2006) is an n -gram language model by Pitman-Yor process which is a generalization of a Dirichlet process. A variable-order Pitman-Yor language model (VPYLM) is an extension of a HPYLM that makes it possible to learn an appropriate context length n of an n -gram. Yoshii and Goto (2011) and Nakano et al. (2015) apply a VPYLM to the chord progression. It is thereby possible to learn an appropriate n -gram length for each chord.

2 Unsupervised melody segmentation using nested Pitman-Yor language model

The musical note sequence \mathbf{s} can be expressed as $\mathbf{s} = s_1 s_2 \cdots s_N$ using musical notes s . When the motif is designated as \mathbf{m} , melody segmentation is to obtain the motif sequence $\mathbf{s} = \mathbf{m}_1 \mathbf{m}_2 \cdots \mathbf{m}_M$. N is the length of the musical note sequence and M is the number of motifs in the musical note sequence.

When the musical note sequence $\mathbf{s} = s_1 s_2 \cdots s_N$ is given, unsupervised melody segmentation is considered as the problem of finding the motif sequence that maximizes the probability $p(\mathbf{m}|\mathbf{s})$ of the motif sequence $\mathbf{s} = \mathbf{m}_1 \mathbf{m}_2 \cdots \mathbf{m}_M$ obtained by dividing the note sequence. The $p(\mathbf{m}|\mathbf{s})$ can be computed by the language model. The model must calculate probabilities for every possible segmentation of the motif to perform a melody segmentation. Using an n -gram language model with a note-level Pitman-Yor process, we can give probabilities for all possible motif segmentations and thus compute the likelihood of the motifs. We can sample the word segmentation on the basis of this probability.

2.1 Modeling of melody using nested Pitman-Yor language model

In this section, the melody is modeled using a NPYLM that is an n -gram language model based on a hierarchical Pitman-Yor (PY) process. The PY process is a stochastic process that generates a discrete probability distribution G , which is similar to a probability distribution G_0 ($G \sim PY(G_0, d, \theta)$). When we have a uni-gram distribution of motifs G_1 , the bi-gram distribution G_2 of motifs will be similar to G_1 . Therefore, we can generate G_2 from a PY process of base measure G_1 ($G_2 \sim PY(G_1, d, \theta)$). The uni-gram motif distribution G_1 can be generated as $G_1 \sim$

$PY(G_0, d, \theta)$. The NPYLM is a hierarchical language model in which the note-level HPYLM is embedded as a base measure of the motif-level HPYLM. For details, see (Teh, 2006).

2.2 Unsupervised melody segmentation and training language model

A straightforward method of melody segmentation is to repeat Gibbs sampling, where every note is sampled with the probability of being a motif boundary, and the language model is updated in accordance with the results of that sampling. We used a sentence-wise Gibbs sampler of word segmentation using efficient dynamic programming (Mochihashi et al., 2009). Sampling a new segmentation, we update the NPYLM by adding a new sentence in accordance with the new segmentation. By repeating this process for all musical pieces in a random order, the melody segmentation and language model are alternately optimized.

The musical note sequence is divided into motifs as follows. $\alpha[t][k]$ is the probability of note sequence $s_1 \cdots s_t$ with the final k characters being a motif.

$$\alpha[t][k] = \sum_{j=1}^{t-k} p(s_{t-k+1}^t | s_{t-k-j+1}^{t-k}) \cdot \alpha[t-k][j] \quad (1)$$

where $\alpha[0][0] = 1$ and $s_i^j = s_i \cdots s_j$. $p(s_{t-k+1}^t | s_{t-k-j+1}^{t-k})$ is obtained by the language model. If $\alpha[t][k]$ can be obtained, we can sample a motif backward. The length of the motif k is sampled from the end of the note sequence to its beginning in accordance with the forward probability $\alpha[t][k]$ (backward sampling). For details, see (Mochihashi et al., 2009).

2.3 Representation of musical note sequence

Musical note sequences can be represented in a number of ways depending on which attributes are used. In this paper, we assume that a musical note in music is like a character in natural language. A melody is considered to be time series data of notes with the properties of various pitches and durations. Therefore, the following representation of musical note sequences with pitch and duration is used.

Pitch-class sequence

A pitch-class sequence considers a melody to be a sequence of pitch classes. The role of each pitch class is assumed to be the same in each key. For

example, the note C in the key of C major and the note D in the key of D major are the same in the sense that they are both the tonic for their key. For this reason, we transpose all the keys to the key of C in advance.

In pitch-class sequences, the octave is ignored, the sharp and the flat are not distinguished, and 12 different symbols are used. There is a total of 13 symbols: 12 symbols for pitch class and 1 for rests.

Pitch-interval sequence

A pitch-interval sequence considers the melody as a sequence of differences between the pitch of the previous note and the current note. We define a pitch-interval sequence on the basis of the assumption that the melody is given meaning by the relative difference in pitch to the previous notes. The Implication-Realization (I-R) model (Narmour, 1990), a music theory that classifies and analyzes melodies, gives an abstract of the melody by focusing on the relationship between the pitches of the notes. The intervals are considered up to two octaves above and below ($-24 \leq d_t \leq 24$). Therefore, the resulting number of symbols is 50 (49 + rest symbol).

Duration sequence

The duration sequence is defined as a sequence of durations focusing only on the duration of the notes in a melody. The durations are limited to the length from a thirty-second note up to two whole notes. We are also able to represent dotted notes and triplets of each note, from thirty-second notes up to whole notes. Rests are treated as a specific symbol with the meaning of a rest.

Compound-representation sequence

The three sequences introduced in the previous section can be combined with one another to form compound representations. First, we combine the pitch-class sequence and duration sequence. Corresponding symbols from the pitch-class sequence and the duration sequence are combined to form a compound representation. We call this the pitch-class and duration sequence (P-D sequence). Second, we combine the pitch-interval sequence and duration sequence. Similarly combining their respective symbols, we form the pitch interval and duration sequence (I-D sequence) Third, the combination of the pitch-class sequence and pitch-interval sequences is called pitch-class and inter-

val sequence (P-I sequence) Finally, we label the combination of all three sequences (pitch-class, pitch interval, and duration) as the P-I-D sequence.

3 Evaluation

In this section, we discuss the characteristics of the melody segmentation obtained with NPYLM, comparing them with the rules and grouping structure of a generative theory of tonal music (GTTM).

3.1 Experimental conditions

To investigate the characteristics of the melody segmentation obtained with NPYLM, we calculate the F-measure for the segments using the ground truth of the grouping structure and each of the rules of the GTTM, although the grouping structure of the GTTM is not necessarily the best for a language model. In this experiment, 300 songs of the GTTM database (Hamanaka) were used as learning data (302 phrases). This dataset consists of monophonic melodies of classical music composed by multiple composers. The total number of notes in the training data set was 12,343, and the average number of notes for each song was 40.9.

The grouping structure of the GTTM represents the cognitive grouping of music experts as they listen to the musical pieces. The sub-rules of the GTTM, the grouping preference rules (GPR), can indicate the candidate boundaries of a group. Each rule does not necessarily coincide with the GTTM grouping structure, but each one mechanically calculates possible boundaries. We compare the segments of the proposed method with the GTTM rules related to the representations of the notes described in Section 3. Specifically, we use GPR 2a, 2b, 3a, and 3d. Given four notes n_1 , n_2 , n_3 , and n_4 , each GPR draws a grouping boundary if the relationship between n_2 and n_3 satisfies the following conditions: $rest_{i-1} < rest_i$ and $rest_i > rest_{i+1}$, where $rest_i$ is the time interval from the beginning to the end of the note (**GPR 2a**); $ioi_{i-1} < ioi_i$ and $ioi_i > ioi_{i+1}$, where ioi_i is the inter-onset interval (**GPR 2b**); $interval_{i-1} < interval_i$ and $interval_i > interval_{i+1}$, where $interval_i$ is the pitch interval (**GPR 3a**); $len_{i-1} = 0$ and $len_i \neq 0$ and $len_{i+1} = 0$, where len_{i-1} is the difference of the duration (**GPR 3d**);

3.2 Experimental results and discussion

Table 1 shows the F-measure of each representation of the musical note sequence. The row (a)

	Representations			GPR 2a (Rest)	GPR 2b (ioi)	GPR 3a (Interval)	GPR 3d (Length)	Grouping Structure
	Pitch	Interval	Duration					
(a)	✓			8.7	21.6	27.7	9.3	29.4
(b)		✓		6.4	21.9	22.6	8.0	23.6
(c)			✓	6.8	18.7	24.3	17.0	34.2
(d)	✓	✓		6.3	18.6	24.9	7.7	21.3
(e)	✓		✓	6.8	21.0	24.6	13.1	28.1
(f)		✓	✓	9.6	19.0	21.8	14.6	24.2
(g)	✓	✓	✓	10.6	17.1	19.5	11.7	21.7

Table 1: F-measure of each representation of the musical note sequence.

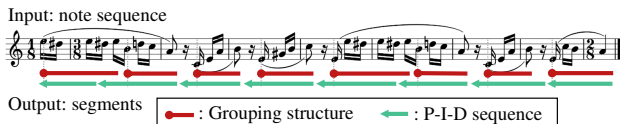


Figure 1: A segmentation result for Bagatelle “Für Elise” WoO.59 (Ludwig van Beethoven) in the GTTM database (No. 3).

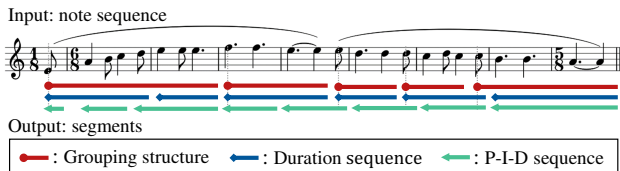


Figure 2: A segmentation result for Má Vlast Moldau (Bedřich Smetana) in the GTTM database (No. 60).

indicates the results of the pitch-class sequence, and the row (d) indicates the results of the P-I sequence. The F-measure for the grouping structure was highest when the duration sequence was used. Figures 1 and 2 show the segmentation results for musical pieces in the GTTM database (No. 3 and No. 60). The lines under the musical score indicate that the notes within the range of the line are in the same grouping.

Regarding GPR 2a, the F-measure was lower than that of the other rules, regardless of which representation was used. The current implementation considers rests to be a special type of note, so distinguishing whether the group boundary is after or before a rest is not possible (see Figure 2). Regarding GPR 3d, F-measure were higher when using a representation related to duration than when using the other representations.

The F-measure for the grouping structure was highest when the duration sequence was used. The grouping structure of the GTTM depends on its metrical structure, such as beats. When the du-

ration sequence is input, we can obtain segments of the rhythmic pattern that occur frequently in a note sequence, because we focus only on the duration, ignoring the pitch completely. In fact, grouping boundaries were drawn more frequently at beat positions when using the duration representation than when using other representations, even though the representation did not explicitly include a metrical structure.

The grouping structure of the GTTM is not necessarily optimal for language models. However, depending on the application, we may have to consider giving information about motifs as prior knowledge and applying semi-supervised learning to obtain the expected melody segmentation. This NPYLM enables semi-supervised melody segmentation.

4 Conclusion

In this study, we performed unsupervised motif segmentation using a Nested Pitman-Yor Language Model. The resulting segments depend on which attributes are used for musical note representation. In the future, we will work on application tasks such as musical structure analysis and representation learning using the obtained segments to verify the usefulness of the segments obtained with the proposed model. We must also consider using other representations, e.g., using abstractions for melody such as I-R model or melodic contour, and explicitly incorporating the metrical structure.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP19J15634.

References

- Emilios Cambouropoulos. 2001. The local boundary detection model (LBDM) and its application in the study of expressive timing. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 232—235.
- Masatoshi Hamanaka. Interactive GTTM Analyzer / GTTM Database Download Page. <http://gttm.jp/gttm/ja/database/>.
- Masatoshi Hamanaka, Keiji Hirata, and Satoshi Tojo. 2017. deepgttm-iii: Multi-task learning with grouping and metrical structures. In *International Symposium on Computer Music Multidisciplinary Research*, pages 238–251.
- Tatsunori Hirai and Shun Sawada. 2019. Melody2vec: Distributed representations of melodic phrases based on melody segmentation. *Journal of Information Processing*, 27:278–286.
- Stefan Lattner, Maarten Grachten, Kat Agres, and Carlos Eduardo Cancino Chacón. 2015. Probabilistic segmentation of musical sequences using restricted boltzmann machines. In *International Conference on Mathematics and Computation in Music*, pages 323–334.
- Fred Lerdahl and Ray S Jackendoff. 1996. *A generative theory of tonal music*.
- Daichi Mochihashi, Takeshi Yamada, and Naonori Ueda. 2009. Bayesian unsupervised word segmentation with nested pitman-yor language modeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP)*, pages 100–108.
- Tomoyasu Nakano, Kazuyoshi Yoshii, and Masataka Goto. 2015. Musical similarity and commonness estimation based on probabilistic generative models. In *2015 IEEE International Symposium on Multimedia (ISM)*, pages 197–204.
- Eugene Narmour. 1990. *The analysis and cognition of basic melodic structures: The implication-realization model*.
- Marcus T Pearce, Daniel Müllensiefen, and Geraint A Wiggins. 2010. *Melodic grouping in music information retrieval: New methods and applications*. Springer.
- Yee Whye Teh. 2006. A hierarchical bayesian language model based on pitman-yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 985–992.
- David Temperley. 2004. *The cognition of basic musical structures*. MIT press.
- Kazuyoshi Yoshii and Masataka Goto. 2011. A vocabulary-free infinity-gram model for nonparametric bayesian chord progression analysis. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*, pages 645–650.

MusicBERT: A Shared Multi-Modal Representation for Music and Text

Federico Rossetto

University of Glasgow
fedingo@gmail.com

Jeff Dalton

University of Glasgow
Jeff.Dalton@glasgow.ac.uk

Abstract

Recent advances in deep learning have led to significant advances in both text and music representations. However, the representations and tasks remain largely separate. Most Music Information Retrieval models focus on either music or text representations but not both. In this work we propose unifying these two modalities in a shared latent space. We propose building on a common framework of Transformer-based encoders for both text and music modalities using supervised and unsupervised methods for pre-training and fine-tuning. We present initial results and key challenges that need to be overcome to make this possible. The result will be a new class of models that are able to perform advanced tasks that span both NLP and music.

1 Introduction

Voice-based conversational agents such as Alexa, Google Assistant, are growing in importance and allow interaction with music systems using natural language. However, current approaches only support interactions with text and metadata (e.g. playing a specific song). This research will enable a multi-modal representation that supports conversation *about* music and its concepts. We propose using new state-of-the-art deep learning models that are capable of producing *joint latent spaces* of both music and text.

We propose a new multi-modal representation model we call *MusicBERT*. It is composed of a set of modality-specific encoders that are then fed to a shared model, based on the Transformer. A key challenge is that audio signals are very long and standard Transformer models are limited in the amount of vectors that they can feasibly and effectively process. As a result, a second layer that encodes music (and its derived concepts) is needed to provide some abstraction with current models.

A high-level view of this model is provided in Figure 1. Similar to how BERT has been adapted for QA and NLP tasks, the proposed architecture can be adapted for music QA tasks (Sutcliffe et al., 2014) using the pre-trained representations.

Although text representations are proven, music remains challenging. In order to have an effective music representation the proposed model must address two key challenges: 1) What is an effective low-level encoding of music that works effectively with Transformers, and 2) What are effective pre-training loss functions for learning music representations that exhibit transfer learning properties. We hypothesize that this requires having the right level of semantic representation.

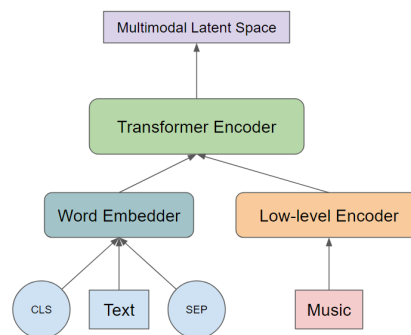


Figure 1: High-level MusicBERT architecture

To train this new multi-modal model we use a standard music datasets the MuMu dataset (Oramas et al., 2017), that maps album reviews to a subset of the Million Song Dataset (Bertinmahieux et al., 2011). These reviews are not very granular, been at album-level, but still provide insightful discussions about the musical content of the tracks contained in the album.

	MFCC	TLM	MTLMR	VGGish	Audioset	MLM	MIM
GTZAN (accuracy)	59.83%	77.60%	65.80%	85.9%	83.00%	73.80%	85.30%
Deezer (r2 score)	7.18%	18.58%	9.60%	20.65%	18.38%	14.58%	16.45%

Table 1: SVM evaluation using the Music Representations extracted

2 Background and Related Work

Textual Representation Learning - The use of Transformers (Vaswani et al., 2017) paired with a language modeling objective is the current state-of-the-art for most NLP tasks. Models such as BERT and similar are effective for NLP tasks and critically they demonstrate strong transfer learning effectiveness (Devlin et al., 2018). We use this as the base for our text-based representations.

Music Representation Recent work by Kim et al. (2019) explores deep representation learning. They apply *multi-task transfer learning* to test the impact on multiple tasks showing an improvement in effectiveness when pre-training on additional external tasks. Similar work on music representations and transfer learning is (Choi et al., 2017). They demonstrate the potential for pre-training on music tagging to create effective *latent representations*. Recent work with Transformers for monophonic music by Huang et al. (2018) begins to address scalability issues. In contrast, we propose representations that generalize to polyphonic music and raw audio.

Multi-modal Representation This work is inspired by multi-modal representations in the field of Computer Vision, and specifically VisualBERT (Li et al., 2019). They develop a model that uses ImageNet concepts to encode the key-points of an image, and train BERT to translate these visual vectors into a textual description. We propose using both encoded audio and audio concepts as a semantic representation. Instead of ImageNet, we use AudioSet (Gemmeke et al., 2017), a concept detector for general audio. Our early results find that more work is needed in developing an ontology specific to music.

3 Method and Preliminary Experiments

In this section we discuss the methods we use including the low-level music encoding and training objective. For low level encoders for our experiments use a word embedding layer as textual encoder, and the VGGish (Hershey et al., 2017) model for the music encoder.

To train this model on music data, we experiment with three different pre-training approaches. A *Masked Language Modeling* algorithm, a *Mutual Information Maximization* algorithm taken from van den Oord et al. (2018) and a standard classification task on the *AudioSet* ontology (Gemmeke et al., 2017). The first algorithm uses a reconstruction loss after masking some of the music vectors. The second instead aims at maximizing the Mutual Information between the music vectors and the multi-modal representations. The last one is just a multi-label classification of sound events.

Results To evaluate the obtained representations, we use standard music tasks, and evaluate the representations using them in an Support Vector Machine (SVM) on the *latent space* for each target, following Choi et al. (2017). We evaluate on the GTZAN (Tzanetakis and Cook, 2002) and Deezer (Delbouys et al., 2018) datasets.

We report the results of our initial experiments in table 1. The results show, VGGish provides a strong representation, and the MIM and MLM training hurt effectiveness. We also provided the results for three baselines. One using the standard MFCCs (Muda et al., 2010), one base on (Choi et al., 2017) (TLM) and one base on (Kim et al., 2019) (MTLMR).

We found that the VGGish encoder output has very limited variability across time and this makes it much more challenging for the model to be effectively trained. Also, there is a blurring effect on the vectors across time caused by the soft-max self-attention. This suggests that using sparse-attention could improve the model effectiveness.

4 Conclusion

We motivate the need for a *multi-modal representation space* and its application to natural language music conversation. We introduce the MusicBERT model to tackle this problem and present preliminary results on standard music tasks. Our goal is to advance the capabilities of current models and enable them to integrate music and text together in new and more effective representations that generalize to complex tasks.

References

- Thierry Bertin-mahieux, Daniel P. W. Ellis, Brian Whitman, and Paul Lamere. 2011. The million song dataset. In *In Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*.
- Keunwoo Choi, George Fazekas, Mark Sandler, and Kyunghyun Cho. 2017. Transfer learning for music classification and regression tasks. In *The 18th International Society of Music Information Retrieval (ISMIR) Conference 2017, Suzhou, China*. International Society of Music Information Retrieval.
- Rémi Delbouys, Romain Hennequin, Francesco Piccoli, Jimena Royo-Letelier, and Manuel Moussallam. 2018. Music mood detection based on audio and lyrics with deep neural net. In *ISMIR*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. [Bert: Pre-training of deep bidirectional transformers for language understanding](#).
- Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. 2017. Audio set: An ontology and human-labeled dataset for audio events. In *Proc. IEEE ICASSP 2017*, New Orleans, LA.
- S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson. 2017. Cnn architectures for large-scale audio classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 131–135.
- Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Curtis Hawthorne, Andrew M. Dai, Matthew D. Hoffman, and Douglas Eck. 2018. [An improved relative self-attention mechanism for transformer with application to music generation](#). *CoRR*, abs/1809.04281.
- Jaehun Kim, Julián Urbano, Cynthia C. S. Liem, and Alan Hanjalic. 2019. [One deep music representation to rule them all? a comparative analysis of different representation learning strategies](#). *Neural Computing and Applications*.
- Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. [Visualbert: A simple and performant baseline for vision and language](#).
- Lindasalwa Muda, Mumtaj Begam, and I. Elamvazuthi. 2010. Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques.
- Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. [Representation learning with contrastive predictive coding](#). *CoRR*, abs/1807.03748.
- Sergio Oramas, Oriol Nieto, Francesco Barbieri, and Xavier Serra. 2017. [Multi-label music genre classification from audio, text, and images using deep features](#).
- Richard FE Sutcliffe, Tim Crawford, Chris Fox, Deane L Root, and Eduard H Hovy. 2014. The c@merata task at mediaeval 2014: Natural language queries on classical music scores.
- G. Tzanetakis and P. Cook. 2002. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#).

Music autotagging as captioning

Tian Cai

The Graduate Center
The City University of New York
New York, 10016, USA
tcai@gradcenter.cuny.edu

Michael I Mandel

Brooklyn College
The City University of New York
New York, 11210, USA
mim@sci.brooklyn.cuny.edu

Di He

The Graduate Center
The City University of New York
New York, 10016, USA
he15810251026@gmail.com

Abstract

Music autotagging has typically been formulated as a multi-label classification problem. This approach assumes that tags associated with a clip of music are an unordered set. With recent success of image and video captioning as well as environmental audio captioning, we propose formulating music autotagging as a captioning task, which automatically associates tags with a clip of music in the order a human would apply them. Under the formulation of captioning as a sequence-to-sequence problem, previous music autotagging systems can be used as the encoder, extracting a representation of the musical audio. An attention-based decoder is added to learn to predict a sequence of tags describing the given clip. Experiments are conducted on data collected from the MajorMiner game, which includes the order and timing that tags were applied to clips by individual users, and contains 3.95 captions per clip on average.

1 Introduction

Music autotagging has been well studied in music information retrieval at ISMIR. From machine learning to deep learning, the community has witnessed progress over the past decade on this task, with new methods (Choi et al., 2016), new model architectures (Yan et al., 2015; Liu and Yang, 2016; Ibrahim et al., 2020; Wang et al., 2019), and new data sets (Law et al., 2009; Bogdanov et al., 2019).

Most studies in content-based autotagging focus on automating the feature extraction to create better representations of music.

What seldom changes, however, is the formulation of the task as a multi-label classification problem (Tsoumakas and Katakis, 2009): treating tags associated with a clip of music as an unordered set. This formulation focuses on correlations between tags, but when a user listens to a clip and provides a sequence of tags, the user expresses his or her listening experience. What is the most “ear-catching” element? What is unexpected? Does this clip feature an instrument or style? These questions cannot be answered under the multi-label classification formulation for music autotagging.

One reason for this formulation is the datasets available for music tagging research, such as MagnaTagATune (Law et al., 2009) and the Million Song Dataset (Bertin-Mahieux et al., 2011). We base the current study on a new analysis of the data collected by the MajorMiner tagging game (Mandel and Ellis, 2008), which includes sequential information. In this game, players supply tags in a particular order and get immediate feedback about the relevance of their tags, further increasing the importance of understanding tag order.

Our switch from multi-label to sequential captions follows similar switches in image and video captioning (Staniute and Šešok, 2019; Chen et al., 2019) and deep learning for acoustic scene and

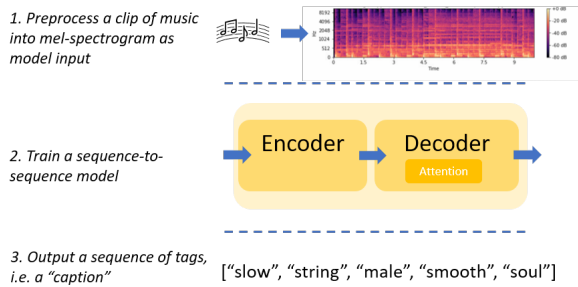


Figure 1: The system uses a sequence-to-sequence model to map mel spectrograms to sequences of tags. The encoder and decoder can be replaced with architectures such as 1D-CNN, 2D-CNN (Choi et al., 2016), MusiCNN (Pons et al., 2017), GRU, LSTM

Encoder	Decoder	Training captions per clip			
		One		Multiple	
		B1	B2	B1	B2
1D CNN	LSTM	9.5	02.2	38.5	38.5
2D CNN	LSTM	10.9	18.3	39.3	48.4
2D CNN	GRU	10.0	19.7	—	—
MusiCNN	LSTM	12.8	23.1	45.8	54.0
MusiCNN	GRU	12.9	23.1	—	—

Table 1: Results on the test set based on the best validation epoch of each model. B1 and B2 stand for BLEU1 and BLEU2, measured in percent.

event captioning. Drossos et al. (2017) presented the first work of audio captioning, focusing on identifying the human-perceived information in a general audio signal and expressing it through text using natural language. The current paper expands this audio captioning approach to the area of music autotagging.

Following these typical captioning models, we use the encoder-decoder architecture with attention mechanism, as shown in Figure 1. We compared three encoders and two decoders, all combined using vanilla attention (Bahdanau et al., 2014). The 2D-CNN for autotagging is from (Choi et al., 2016) and the MusiCNN is from (Pons et al., 2017). We remove the final prediction layer and use the final embedding as the feature fed into the decoder. For the decoder, we compare the most common two choices in image captioning and video captioning: RNN-GRU and LSTM. All models are trained using teacher forcing with cross-entropy of the predicted tag as their loss. It is not attempted in this paper to propose new captioning architecture but to draw awareness of the potential and benefits to re-define music autotagging task leveraging the advancement in NLP.

2 Related Work

Several papers have explored the co-occurrence relationships between tags: Miotto et al. (2010) present one of the early works that explicitly used tag co-occurrence modeled by a Dirichlet mixture. Shao et al. (2018) modeled the tag co-occurrence pattern of a song via Latent Music Semantic Analysis (LMSA). Larochelle et al. (2012); Mandel et al. (2010, 2011a,b) utilized tags alone to build a conditional restricted boltzmann machine and hence demonstrated the value of tag-tag relationships in predicting tags.

Recent works such as (Choi et al., 2018) discussed the effect of tags from the perspective of mislabeling under the theme of multi-label classification.

Following (Drossos et al., 2017), Gharib et al. (2018) also first applied domain adaptation techniques as used in NLP to scene classification. Drossos et al. (2019) added language modeling for sound event detection. Ikawa and Kashino (2019) used a captioning model to describe environmental audio. They proposed an extension to the standard sequence-to-sequence model in the captioning task by adding a controllable parameter, specifying the amount of context to provide in the caption.

Multi-label classification is a challenging and important task not only in music information retrieval but also in field such as document categorization, gene function classification and image labeling. In image labeling, Wang et al. (2016) has demonstrated the effectiveness of using RNNs to learn correlation among labels. However, what we propose is not only to learn the label correlations, but also capture user experience with music from the order of tags.

3 Dataset: MajorMiner

Guided by the goal of multi-label classification, most datasets do not retain or make available information about the ordering of tags by users. MajorMiner (Mandel and Ellis, 2008) is a web-based game¹ that naturally collects this information. Participants describe 10-second clips of songs and score points when their descriptions match those of other participants. Users are given the freedom to use any tag they want, but the rules were designed to encourage players to be thorough and the clip length was chosen to make judgments objective

¹<http://majorminer.org/>

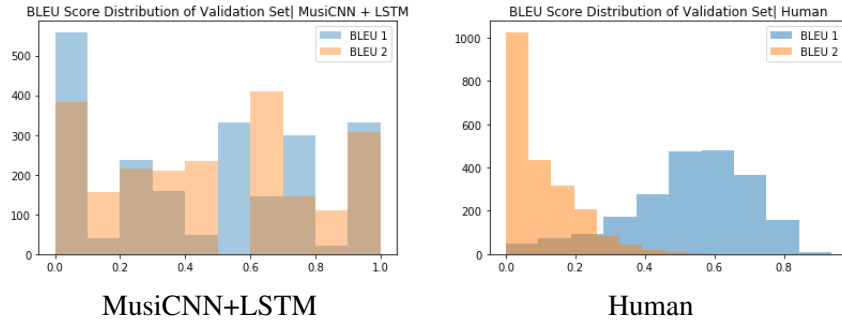


Figure 2: Bleu score distribution of validation data set using (a) the best epoch of MusiCNN + LSTM model and (b) inter-annotator BLEU score, both with multiple captions per clip.

and specific.

As required by the captioning task, one sample consists of a pair consisting of one audio clip and one corresponding caption provided by one user. The MajorMiner game is designed to collect sequences of tags describing a clip one tag at a time from a user. A sequence of tags is collected, which are ordered by time stamps, and act as a caption. By design, one clip is frequently heard by several different users (this is the only way that any of them may score it). Hence, one clip will receive several captions. This fits into the multi-reference scenario (Papineni et al., 2002) that is often encountered in NLP, for example, in machine translation, where one source sentence has many valid translations into another language.

Caption data is pre-processed through case folding, removal of punctuation, and porter stemming. Sequences of tags, which get validity confirmed, are normalized and canonicalized. The longest tag sequence for a single clip is 30 tags. The total tag vocabulary is 984. Clips are randomly partitioned into train/valid/test set in the ratios of 75% – 15% – 10%.

Log mel spectrograms with 96 mel bins are used as input for all models. With sample rate 12,000 Hz, the length of the FFT window is 512 samples (42 ms), and 256 samples between successive frames (21 ms). Each 10-second clip becomes a 469×96 matrix.

4 Experiments and Analysis

A series of experiments is carried out, pairing three encoders, 1D CNN, 2D CNN, MusiCNN, and two decoders, GRU and LSTM, under two settings, multiple captions per clip and one caption per clip, as shown in Table 1. BLEU1 and BLEU2 are used to evaluate each model’s ability to capture tag orders.

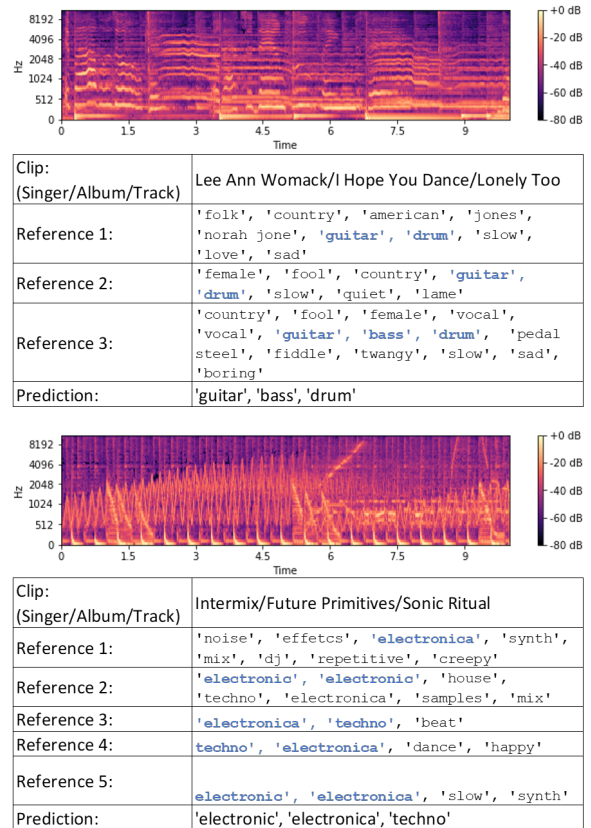


Figure 3: Prediction examples from the best epoch of the MusiCNN + LSTM model. Examples are from the validation set.

Other metrics that are standard in music autotagging will be used in future work. The reason to create two caption settings is that, while there are multiple captions per clip, this potentially complicates training a model. Thus, our initial experiments are restricted to a single caption per clip where that caption is selected at random from those applied to that clip. In the multi-caption-per-clip scenario, if a clip received four captions, it is presented in four caption-clip pairs in training, one with each caption. Yet, in the calculation of BLEU scores, all four reference sequences are used for the one clip.

MusiCNN provides both a waveform-based front end and spectrogram-based front end. We use the spectrogram-based front end to make fair comparisons with other encoders. MusiCNN used in this paper has the same configuration as in music autotagging papers (Pons et al., 2017). The 2D CNN used in this paper has six layers of 2D convolution, each followed by batch normalization and 2D max pooling. We also compare a 1D-CNN as an encoder in an attempt to retain more temporal information. This is out of the consideration that some tags appear only at some time steps. In the 1D-CNN, the frequency axis of the mel-spectrogram is taken as the “channel” so that convolutions are computed along the time axis only. The 1D CNN used in this paper has six layers of 1D convolution, each followed by batch normalization. Both 2D and 1D CNN use 256 filters and ReLu at each convolution layer. Both GRU and LSTM decoders have only one layer of RNN followed by two fully connected layers. All models use sparse categorical cross-entropy as loss function and Adam as optimizer.

We create a naïve baseline for the multi-captioning setting by predicting the top k most popular tags for all clips. This evaluates the amount of information our models are learning beyond frequency. Figure 4 shows that this baseline’s best performance is to use top three most frequent tags, which achieves a BLEU1 of 0.49 and a BLEU2 of 0.086. This BLEU1 is comparable to our model, but the BLEU2 is much lower. A better baseline for BLEU2 might apply the most common sequence of bigrams. This will be evaluated in future work.

5 Results

The upper bound on the performance of our model is the inter-annotator agreement. Thus, we measure the BLEU score of our ground truth captions

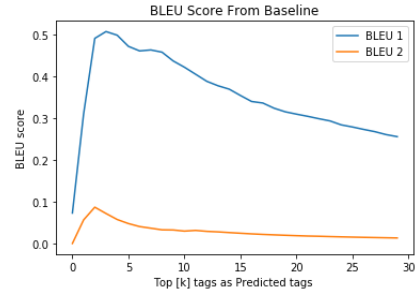


Figure 4: Average validation BLEU score for baseline selecting the k most popular tags for all clips with multiple captions per clip.

with relation to the other captions of the same clip. We find that the average BLEU1 in this case is 0.53 and the average BLEU2 is 0.09. Surprisingly, this is far below that of our best model MusiCNN+LSTM. Beyond the mean, the distribution of inter-annotator BLEU scores is shown in Figure 2(b). Comparing with Figure 2(a), the BLEU score distribution for human tag sequences is smoother and unimodal. This helps to understand why our training/validation BLEU score curve improves very slowly.

To further analyze our results, Figure 2(a) shows a histogram of BLEU1 and BLEU2 scores for the MusiCNN+LSTM model. As can be seen, there is high variability in performance across clips with some having very high scores and some very low scores. This phenomenon may be because tags such as “guitar” are quite frequent and heavily influence the model training, leading to better performance on samples where those tags are relevant. Dealing with imbalances in word frequencies is a common issue in NLP, but we leave it for future work.

Figure 3 shows example annotations, spectrograms, and predictions from the MusiCNN+LSTM model on two example clips. It shows that the model is able to capture general genre information but lacks the nuance of the human annotations.

6 Conclusion and Future Work

The paper demonstrates the promise of formulating music autotagging as a captioning task. It also opens up new possibilities for music autotagging. More advanced NLP techniques such as Transformers (Vaswani et al., 2017; Zhou et al., 2018; Yu et al., 2019) and Masked Language Model pre-training (Devlin et al., 2018) could be utilized to enhance the performance of a language model for

music. There is still more information in the sequence of tags that are applied to a clip that we are not using, such as the temporal locality of tags such as “clap.” As pointed out in the recent audio captioning work (Çakır et al., 2020), the distribution of words in captions is a significant challenge. Future work will also address the issue of very frequent yet less useful tags.

References

- Dzmitry Bahdanau, Kyunghyun Cho, and Y. Bengio. 2014. Neural machine translation by jointly learning to align and translate. *ArXiv*, 1409.
- Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. 2011. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*.
- D Bogdanov, M Won, P Tovstogan, A Porter, and X. Serra. 2019. The mtg-jamendo dataset for automatic music tagging.
- Emre Çakır, Konstantinos Drossos, and Tuomas Virtanen. 2020. Multi-task regularization based on infrequent classes for audio captioning. *arXiv preprint arXiv:2007.04660*.
- Shaoxiang Chen, Ting Yao, and Yu-Gang Jiang. 2019. *Deep learning for video captioning: A review*. pages 6283–6290.
- Keunwoo Choi, George Fazekas, and Mark Sandler. 2016. Automatic tagging using deep convolutional neural networks.
- Keunwoo Choi, György Fazekas, Kyunghyun Cho, and Mark Sandler. 2018. *The effects of noisy labels on deep convolutional neural networks for music tagging*. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2:139–149.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.
- Konstantinos Drossos, Sharath Adavanne, and Tuomas Virtanen. 2017. Automated audio captioning with recurrent neural networks. In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 374–378. IEEE.
- Konstantinos Drossos, Shayan Gharib, Paul Magron, and Tuomas Virtanen. 2019. *Language modelling for sound event detection with teacher forcing and scheduled sampling*. *CoRR*, abs/1907.08506.
- Shayan Gharib, Konstantinos Drossos, Emre Cakir, Dmitriy Serdyuk, and Tuomas Virtanen. 2018. Un-supervised adversarial domain adaptation for acoustic scene classification. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, pages 138–142.
- Karim Ibrahim, Jimena Royo-Letelier, Elena Epure, Geoffroy Peeters, and Gael Richard. 2020. *Audio-based auto-tagging with contextual tags for music*. pages 16–20.
- Shota Ikawa and Kunio Kashino. 2019. Neural audio captioning based on conditional sequence-to-sequence model. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pages 99–103, New York University, NY, USA.
- Hugo Larochelle, Michael Mandel, Razvan Pascanu, and Y. Bengio. 2012. Learning algorithms for the classification restricted boltzmann machine. *The Journal of Machine Learning Research*, 13:643–669.
- Edith Law, Kris West, Michael Mandel, Mert Bay, and J. Downie. 2009. Evaluation of algorithms using games: The case of music tagging. pages 387–392.
- Jen-Yu Liu and yi-hsuan Yang. 2016. *Event localization in music auto-tagging*. pages 1048–1057.
- Michael Mandel, Douglas Eck, and Y. Bengio. 2010. Learning tags that vary within a song. pages 399–404.
- Michael Mandel, Razvan Pascanu, Douglas Eck, Y. Bengio, Luca Aiello, Rossano Schifanella, and Filippo Menczer. 2011a. *Contextual tag inference*. *ACM Transactions on Multimedia Computing, Communications, and Applications - TOMCCAP*, 7S:1–18.
- Michael Mandel, Razvan Pascanu, Hugo Larochelle, and Yoshua Bengio. 2011b. *Autotagging music with conditional restricted boltzmann machines*.
- Michael I. Mandel and Daniel P. W. Ellis. 2008. *A web-based game for collecting music metadata*. *Journal of New Music Research*, 37(2):151–165.
- Riccardo Miotto, Luke Barrington, and Gert Lanckriet. 2010. Improving auto-tagging by modeling semantic co-occurrences. pages 297–302.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei Jing Zhu. 2002. *Bleu: a method for automatic evaluation of machine translation*.
- Jordi Pons, Oriol Nieto, Matthew Prockup, Erik Schmidt, Andreas Ehmann, and Xavier Serra. 2017. End-to-end learning for music audio tagging at scale.

- Xi Shao, Zhiyong Cheng, and Mohan Kankanhalli. 2018. [Music auto-tagging based on the unified latent semantic modeling](#). *Multimedia Tools and Applications*, 78.
- Raimonda Staniute and Dmitrij Šešok. 2019. A systematic literature review on image captioning. *Applied Sciences*, 9(10.3390/app9102024):2024.
- Grigorios Tsoumakas and Ioannis Katakis. 2009. [Multi-label classification: An overview](#). *International Journal of Data Warehousing and Mining*, 3:1–13.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#).
- Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. 2016. [Cnn-rnn: A unified framework for multi-label image classification](#). In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2285–2294.
- Qianqian Wang, Feng Su, and Yuyang Wang. 2019. [A hierarchical attentive deep neural network model for semantic music annotation integrating multiple music representations](#). pages 150–158.
- Q. Yan, C. Ding, J. Yin, and Y. Lv. 2015. [Improving music auto-tagging with trigger-based context model](#). In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 434–438.
- Jun Yu, Jing Li, Zhou Yu, and Qingming Huang. 2019. [Multimodal transformer with multi-view visual representation for image captioning](#). *IEEE Transactions on Circuits and Systems for Video Technology*, PP:1–1.
- Luwei Zhou, Yingbo Zhou, Jason Corso, Richard Socher, and Caiming Xiong. 2018. [End-to-end dense video captioning with masked transformer](#). pages 8739–8748.

Comparing Lyrics Features for Genre Recognition

Maximilian Mayerl* Michael Vötter* Manfred Moosleitner Eva Zangerle

Universität Innsbruck

Department of Computer Science

Technikerstraße 21a, 6020 Innsbruck, Austria

firstname.lastname@uibk.ac.at

Abstract

In music information retrieval, genre recognition is the task of automatically assigning genre labels to a given piece of music. Approaches for this typically employ machine learning models trained on content features extracted from the audio. Relatively little attention has been given to using textual features based on a song’s lyrics to solve this task. We therefore investigate how well such lyrics features work for the task of genre recognition by training and evaluating models based on various sets of well-known textual features computed on song lyrics. Our results show that textual features produce accuracy scores comparable to audio features. Further, we see that audio and textual features complement each other well, with models trained using both types of features producing the best accuracy scores. To aid the reproducibility of our results, we make our code publicly available.

1 Introduction

Genre recognition is the task of automatically detecting the genre(s) of a given piece of music and often relies on audio features describing the song, including spectral, rhythmic, and tonal features. On the other hand, comparatively little work exists on the effectiveness of lyrics features to build genre recognition models, especially looking into different types of lyrics features.

Ying (2012) looked into using part-of-speech (POS) information calculated on lyrics to detect genre and mood labels for songs. They used a dataset of 600 songs and trained three different machine learning models—k-nearest neighbour, naive Bayes, and support vector machines (SVM)—to determine how well these POS features perform for genre and mood detection. Tsaprasinos (2017) used a hierarchical recurrent

neural network model taking word embeddings of raw lyrics as input to predict genres. They performed experiments with 20 and 117 different genres on a dataset of around 450,000 songs and achieved accuracies of 46.42% and 49.50%, respectively. Fang et al. (2017) investigated the effectiveness of various textual features for genre recognition and release date estimation, focusing on discourse-based features, as opposed to features which only take into account single sentences. They found that discourse-based features were effective for genre recognition. Mayer et al. (2008) used features capturing the song’s rhythm and features reflecting the structure and statistics of rhymes, tf-idf, and POS, to train kNN, naive Bayes, and SVM algorithms on two datasets with 600 and 3010 songs, respectively. McKay et al. (2010) used a meta-learning based algorithm to predict the genre of songs, by training the algorithm on individual and on combinations of symbolic, lyrical, audio, and cultural features of 250 songs. The results of Mayer et al. and McKay et al. suggest that combining feature groups can improve results compared to training on individual feature groups.

In this paper, we perform a study on the effectiveness of various widely used textual features, computed on song lyrics, for genre recognition. We use the ALF-200k dataset (Zangerle et al., 2018) of songs with English lyrics and add genre information to the songs contained in that dataset via the Last.fm API¹. We then train machine learning models on the genre recognition task using various sets of widely used textual features and multiple different machine learning models, to determine how well the different types of features perform for genre recognition.

*Authors contributed equally to this work.

¹<https://www.last.fm/api/>

Genre	#Songs	Genre	#Songs
alternative	6,828	jazz	1,147
blues	1,101	metal	2,542
country	1,861	pop	7,861
dance	1,539	punk	1,564
electronic	2,677	rap	1,662
funk	791	rnb	1,556
hip hop	2,459	rock	17,234
indie	7,405	soul	2,710

Table 1: Number of songs per genre in our dataset.

2 Dataset

To perform our experiments, we require a collection of song lyrics for a sufficiently large set of songs. Our choice fell on the ALF-200k dataset (Zangerle et al., 2018). This dataset provides lyrics-based textual features for a collection of around 200,000 songs. Since we also require raw song lyrics for our experiments, we downloaded those using the code provided by ALF-200k. Further, we removed duplicates from the ALF-200k dataset based on artist and title where we kept the first occurrence of these songs.

In addition to lyrics, we also need genre labels for the songs in our dataset. As ALF-200k does not provide those, we obtained these via the Last.fm platform. For this, we used the API to search for the songs in our dataset based on their artist and track names, and retrieved the assigned tags from Last.fm. To get genre labels from those tags, we take the 40 most common tags and then only keep tags that represent genres. Additionally, we manually group sub-genres into parent genres based on suffix (e.g., if a song is tagged as *alternative rock*, we assign the genre label *rock* to it), resulting in 16 different genres as shown in Table 1.

Ultimately, we end up with a dataset consisting of 35,045 songs (songs which we did not find on Last.fm were removed) and 16 genre labels. Additionally, the dataset also contains 50 pre-computed textual features per song, taken from ALF-200k, and 10 audio features. The number of songs per genre in our dataset can be seen in Table 1. Note that, as the genre labels are not mutually exclusive, multiple genres can be assigned to a single song.

3 Methods and Experiments

To determine the effectiveness of different types of textual features for the task of genre recogni-

tion, and the extent to which those features complement each other, we performed a range of experiments using different types of features and machine learning models. In this section, we will first provide details about the used features and machine learning models, and then elaborate on the experimental setup.

3.1 Features

As mentioned in Section 2, the ALF-200k dataset contains 50 pre-computed textual features and 10 audio features. We used those features and grouped them into five categories (the exact list of features for each category can be found in the code²):

- **rhymes:** This group contains features describing the rhymes contained in the song lyrics. This includes features like rhymes per line, rhyme density, number of perfect rhymes etc. Those features were taken from ALF-200k, for which they were computed using the rhyme analyzer tool of Hirjee and Brown (2010). In total, there are 15 features in this group.
- **statistical:** This group contains statistical text features computed over the full text of a song’s lyrics. Examples of features in this group include token count, line count, stop-word ratio, proportion of novel words, ratio of lines that are repeated, etc. In total, there are 31 features in this group.
- **statistical_time:** This group contains statistical text features that are computed over a song’s duration. Overall, there are three features in this group: words per minute, characters per minute, and lines per minute.
- **explicitness:** This group contains only a single feature, which is a binary label, as given by the Spotify API³, indicating whether a song’s lyrics are explicit or not.
- **audio:** This group consists of ten high-level audio features such as acousticness, danceability or tempo, taken from the ALF-200k dataset, for which they were obtained via the

²<https://github.com/dbis-uibk/NLP4MusA2020>

³<https://developer.spotify.com/documentation/web-api/>

Spotify API. We use these features for an audio-based baseline for our experiments.

In addition to those five feature groups containing features stemming from the ALF-200k dataset, we computed the following two additional types of features using the raw lyrics texts of the songs in the dataset with no further pre-processing:

- **tf-idf**: We computed tf-idf vectors over n-grams (uni- to trigrams) on the raw lyrics texts. To limit the length of the resulting feature vector, we only considered the top 2,000 most frequent n-grams.
- **lda**: We also computed feature vectors using Latent Dirichlet Allocation (LDA) (Blei et al., 2003), to capture the topics expressed in the lyrics. We used topic vectors with 25 components (i.e., topics) for our experiments.

3.2 Models

We employed multiple different machine learning algorithms to determine how well the feature groups described in Section 3.1 perform for inferring a song’s genres. This was done to be able to quantify how well the features perform, independent of the concrete machine learning model used. In total, we used five different models: k-nearest neighbors (kNN), random forests (RF), forests of extremely randomized trees (ET) (Geurts et al., 2006), support vector machines (SVM), and a self-normalizing neural network model (NN) (Klambauer et al., 2017). For all those models, we performed grid searches with five-fold cross validation to find well-performing parameter settings.

For kNN, RF, ET, and SVM, we used the implementation provided in scikit-learn⁴. For the neural model, we used a simple feed-forward architecture with two hidden layers (both with either 32 or 64 units), both of which use SELU activation and an alpha dropout of 0.1, as described by Klambauer et al. (2017). The neural model was implemented using TensorFlow⁵.

3.3 Experimental Setup

As a first step, we computed a random baseline, which assigns every song to every genre with uniform probability (i.e., every given genre has 50% probability of being assigned to a given song).

⁴We used version 0.23.1 for our experiments.

⁵We used version 2.2.0 for our experiments.

Following that, we calculated a baseline by training and evaluating all of the models described in Section 3.2 on the *audio* feature group. This makes it possible to compare the performance of models using only textual features to models using only audio features.

Then we evaluated the same models on all other text-based feature groups described in Section 3.1 individually for every feature group. Lastly, since we were also interested in seeing how well the textual features complement each other, and how well textual features can complement audio features, we evaluated our machine learning models on (1) a combination of all text-based features groups, and (2) a combination of all text-based feature groups plus *audio*. As mentioned before, all our experiments were performed using five-fold cross validation using the provided methods of scikit-learn.

4 Results

For our evaluation, we used the F_1 score, and since our task is a multi-label problem with an imbalanced distribution of labels, we used macro-averaging to calculate the reported scores.

A summary of the results is given in Table 2, where we depict the F_1 score for every combination of feature group and machine learning model. In every case the reported results are taken from the best model parametrization identified by the grid search. We used the score for the best performing model since we want to determine the effectiveness of the textual features, independent of the concrete machine learning model.

Comparing the random baseline to the results of the other experiments, we observe that every single feature group outperforms the random baseline. We conclude that every proposed feature group (textual feature groups or audio features) carries useful information. The smallest difference between the baseline and an actual feature group is found for both *rhymes* and *explicitness*, which both have a best F_1 score of 0.179, compared to 0.156 for the baseline.

We also observe that only one textual feature group achieved better results than *audio* features: *tf-idf*, with a maximum F_1 score of 0.310 compared to 0.277 for the models using audio features. The next best singular textual feature groups are *lda* and *statistical*, with an almost identical score of 0.233 and 0.231, respectively. The remaining textual feature groups (*rhymes*, *statistical_time*,

Feature Group	Extra Trees	Neural Network	Random Forest	SVM	kNN	Best
uniform random	—	—	—	—	—	0.156
audio	0.187	0.250	0.200	0.191	0.277	0.277
rhymes	0.107	0.105	0.116	0.179	0.157	0.179
statistical	0.166	0.199	0.169	0.193	0.231	0.231
statistical.time	0.140	0.123	0.131	0.194	0.176	0.194
explicitness	0.077	0.089	0.077	0.179	0.063	0.179
tf-idf	0.141	0.310	0.152	0.211	0.203	0.310
lda	0.177	0.171	0.183	0.219	0.233	0.233
combined	0.156	0.334	0.162	0.214	0.237	0.334
combined + audio	0.155	0.371	0.166	0.220	0.233	0.371

Table 2: Summary of our experimental results. For every feature group or a combination thereof, and every machine learning model, we report the F_1 score of the best parametrization found by the grid search.

and *explicitness*) also showed comparable performances, with scores of 0.179, 0.194, and 0.179, respectively. We can also see that combining all textual features (*combined*) leads to improved performance compared to the best performing singular textual feature group (0.334 compared to 0.310 for *tf-idf*). From this, we can conclude that the different textual features capture orthogonal information, and models can benefit from using all of them as their input. Further, adding audio features to the textual features (*combined + audio*) again improves performance. This implies that textual and audio-based features capture orthogonal information and therefore, complement each other.

Lastly, inspecting the performance of individual machine learning models, we observe that for the models using singular feature groups, the best performing machine learning models change between feature groups, with kNN producing the best results for *audio*, *statistical*, and *lda*, SVM producing the best results for *rhymes*, *statistical.time*, and *explicitness*, and the neural model producing the best results for *tf-idf*. For the combined feature sets, the best results are produced by the neural models. As *tf-idf* contains the largest number of features, this could imply that the neural model is best at handling a large number of (sparse) features.

5 Conclusion

In this paper, we investigated the effectiveness of textual features based on song lyrics for genre recognition. We found that such features can be used to train machine learning models which significantly outperform a random baseline. We also

found that at least one type of textual feature, namely *tf-idf*, outperforms a simple set of audio-based descriptors.

We further looked into how well different textual features complement each other, and how well they combine with audio features. We found that combining all the textual feature types leads to a significantly increased accuracy, and that adding audio-based features boosted accuracy even more.

In future work, further combinations of features, feature groups, and models should be considered. As we would have expected an increase in performance for all models when joining *combined* with *audio*, we were surprised to only see that effect for the neural network model. To get a deeper understanding for why this happens, it is necessary to analyze the performance in more detail. For example it seems to be valuable to analyze the expressiveness of single features of individual feature groups as well as different combinations thereof. Further, since our results suggest that different machine learning models perform best for different types of features, investigating the use of an ensemble of models might be interesting.

Acknowledgments

We would like to thank Prof. Günther Specht for providing the necessary infrastructure to perform the research reported in this paper.

References

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022.

- Jiakun Fang, David Grunberg, Diane T Litman, and Ye Wang. 2017. Discourse analysis of lyric and lyric-based classification of music. In *Proc. 18th International Society for Music Information Retrieval Conference*, pages 464–471. International Society for Music Information Retrieval.
- Pierre Geurts, Damien Ernst, and Louis Wehenkel. 2006. Extremely randomized trees. *Machine learning*, 63(1):3–42.
- Hussein Hirjee and Daniel G Brown. 2010. Rhyme analyzer: An analysis tool for rap lyrics. In *Proc. 11th International Society for Music Information Retrieval Conference*. International Society for Music Information Retrieval.
- Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. 2017. Self-normalizing neural networks. In *Proc. 31st International Conference on Neural Information Processing Systems*, pages 972–981.
- Rudolf Mayer, Robert Neumayer, and Andreas Rauber. 2008. [Combination of audio and lyrics features for genre classification in digital audio collections](#). In *Proceedings of the 16th ACM International Conference on Multimedia, MM '08*, page 159–168. Association for Computing Machinery.
- Cory McKay, John Ashley Burgoyne, Jason Hockman, Jordan BL Smith, Gabriel Vigliensoni, and Ichiro Fujinaga. 2010. Evaluating the genre classification performance of lyrical features relative to audio, symbolic and cultural features. In *Proc. of 11th International Society for Music Information Retrieval Conference*, pages 213–218. International Society for Music Information Retrieval.
- Alexandros Tsaptsinos. 2017. Lyrics-based music genre classification using a hierarchical attention network. In *Proc. 18th International Society for Music Information Retrieval Conference*.
- Teh Chao Ying, Shyamala Doraisamy, and Lili Nurliyana Abdullah. 2012. Genre and mood classification using lyric features. In *2012 International Conference on Information Retrieval & Knowledge Management*, pages 260–263. IEEE.
- Eva Zangerle, Michael Tschuggnall, Stefan Wurzinger, and Günther Specht. 2018. [Alf-200k: Towards extensive multimodal analyses of music tracks and playlists](#). In *Advances in Information Retrieval - 39th European Conference on IR Research, ECIR 2018*, pages 584–590, Cham. Springer.

Classification of Nostalgic Music Through LDA Topic Modeling and Sentiment Analysis of YouTube Comments in Japanese Songs

Kongmeng Liew¹, Yukiko Uchida², Nao Maeura¹, and Eiji Aramaki³

¹Graduate School of Human and Environmental Studies, Kyoto University, Japan

²Kokoro Research Center, Kyoto University, Japan

³Graduate School of Science and Technology, Nara Institute of Science and Technology, Japan

¹{ryuu.konmin.56x, maeura.nao.35x}@st.kyoto-u.ac.jp

²uchida.yukiko.6m@kyoto-u.ac.jp

³aramaki@is.naist.jp

Abstract

Nostalgia has been defined as a bittersweet, social emotion, that is often induced through music. In this paper, we examine how these may be expressed in Japanese YouTube comments of nostalgic (mid-2000s) and non-nostalgic (recent) songs (music videos). Specifically, we used sentiment analysis and Latent Dirichlet Allocation (LDA) topic modeling to examine emotion word usage and broader themes across comments. A gradient boosted decision tree classifier was then able to classify nostalgic and non-nostalgic music videos above chance level. This suggests that analyses on video/music comments may be a possible method to quantify expressions of listener emotions, and categorise musical stimuli.

1 Introduction

The last decade has seen a sharp increase of nostalgia-related research in the psychology-emotion literature. Nostalgia has been defined primarily as a self-relevant emotion, in that the self is experienced through narratives of autobiographical events. Yet, it is also a social emotion, in that these narratives also involve memories of social interaction, ultimately fostering a sense of social connectedness (Tilburg et al., 2017, 2018; Reid et al., 2015; Vess et al., 2012). It has been characterized as a bittersweet experience, mixing feelings of pleasantness with appraisals of irretrievable loss (Tilburg et al., 2018), particularly in reflecting and savouring past social experiences (Biskas et al., 2019). In music, it is often induced by sadness (Taruffi and Koelsch, 2014), and is stronger for music associated with reminiscence bumps (i.e., disproportionately recalled for events in late adolescence and early childhood, Krumhansl and Zupnick, 2013)

In this paper, we propose that since nostalgia has such distinct elicitors and appraisals, au-

tomatic classification of nostalgic popular songs should be possible by analysing listener responses. Here, we operationalise these responses as comments on music videos in YouTube. We first use unsupervised learning (topic modelling) and sentiment analysis to quantify comments into features, and use supervised learning (gradient boosted decision trees (GBDT, Friedman, 2001)) to classify comments belonging to nostalgic (old) music videos, or non-nostalgic (recent) music videos based on the identified topics and sentiment categories.

2 Related Work

In the field of Music Information Retrieval, social media (Twitter posts) has been previously used in the context of music entity recognition. Porcaro and Saggion (2019) developed a method of identifying aspects of broadcast classical music through corresponding Twitter activity. For nostalgia and music, Timoney, Davis and Raj (2018) mined 556 comments from YouTube music videos from British hit songs between 1960 – 1970, and found that nostalgic comments could be classified with 86% accuracy (from non-nostalgic comments). This mirrors research from Davalos and colleagues (2015), who found distinctive characteristics of nostalgic posts on Facebook: nostalgic posts tended to have more reflective and emotional content, tinged with mixed positive and negative elements. Our analysis adds to this body of research, in that we seek to use classify comments belonging nostalgic and non-nostalgic music videos in Japanese.

3 Method

We first conducted an online pilot study, where $N(\text{participants}) = 342$ participants rated one randomly selected song (out of a total set of 20 songs)

Track	Artist	Condition	Year
No More Cry	D-51	Nostalgic	2005
Kibun Jou Jou	Mihimaru GT	Nostalgic	2006
Goodbye Days	YUI	Nostalgic	2007
Sakura	Naotaro Moriyama	Nostalgic	2002
Wataridori	[Alexandros]	Non-Nostalgic	2015
Stay Tune	Suchmos	Non-Nostalgic	2017
Chocho Musubi	Aimer	Non-Nostalgic	2016
Himawari no Yakusoku	Motohiro Hata	Non-Nostalgic	2015

Table 1: List of songs per condition and release year. YouTube IDs for each video are available in our online supplementary material.

on felt nostalgia. Each song received ratings from approximately 10 participants. From this, we selected 8 songs that scored the highest (and lowest) on felt nostalgia via a single-item, 7-point Likert scale. We defined nostalgic songs as songs that were popular within Japan in the mid-2000s, that likely induced nostalgia for those aged around 25-35. Non-nostalgic songs were recently popular songs released within the last 5 years (see Table 1). We then identified 37 YouTube videos that corresponded to these 8 songs and obtained a list of all YouTube comments through the YouTube API via the ‘tubeR’ wrapper in R (Sood, 2019). To ensure the overall representativeness of our study, this excluded videos that had less than 50000 views, were not in Japanese, and collaboration videos. Additionally, we filtered out exceptionally short comments (that were deemed unsuitable for analysis), by excluding the shortest (25th percentile) comments from the dataset. We also removed all alphanumeric characters and non-Japanese text, and tokenised the remaining Japanese comments through the RMeCab (Ishida, 2018) wrapper for the MeCab software (Kudo, 2005). This converted Japanese terms and phrases into their simplest (plain) forms, allowing for more consistency in both topic modelling, and matching with the emotion dictionary. We obtained a final N(comments) = 710 (Nostalgic = 324, Non-Nostalgic = 386). We obtain scores for emotion tags and LDA posterior probabilities for all comments, and divided them into training (0.6) and testing (0.4) sets. We used GBDT (‘gbm’ package; Greenwell et al., 2019, using the ‘caret’ wrapper; Kuhn, 2019) to classify them as nostalgic or non-nostalgic, and use partial dependency plots and variable importance measures to interpret these results. Note that all analyses were conducted in R (R Core Team, 2019).

3.1 Text Analyses

For the sentiment analyses, we used the JIWC emotion dictionary (Shibata et al., 2017). This matched words to 7 emotion categories (happy, sad, anger, surprise, trust, anxiety, hate/disgust; fear was excluded) based on a translation of Pluchik’s (1980) emotion wheel, and scores for each comment (S_{ij}) were a ratio of number of emotion terms in each category (W_{ij}), to the total number of terms (tokens; W_{i*}) in each comment:

$$S_{ij} = \frac{W_{ij}}{W_{i*}} \log(W_{ij} + 1) \quad (1)$$

For topic modelling, in order to reduce the bias caused by human supervision, this study employed the unsupervised Latent Dirichlet Allocation (LDA; Blei et al., 2003). LDA identifies latent topics from documents (in this case, comments), through modelling the probabilistic distribution of topics in a document, and words in topics. LDA topic modelling with Gibbs sampling was conducted used the ‘topicmodels’ package (Grun and Hornik, 2011), and the number of topics was determined used the method described in Griffiths and Steyver (2004), which uses the posterior probability for each model (with varying numbers of topics) from all words in the corpus of YouTube comments. For each document, we used the resultant probability distribution for each topic as features in a classification model alongside the JIWC emotion categories.

4 Results

A total of 14 topics were identified (see Figure 1; a list of top terms for all topics are available in our online supplementary material: <https://osf.io/52abe/>). These were combined with scores from the 7 JIWC emotion categories and word count, and fitted in a GBDT classification model, for a total of 22 features. Parameter se-

lection for the model was determined through 12-fold cross validation on the training set, resulting in $n(\text{trees}) = 50$ and interaction depth = 1. An overall modest accuracy score of $\text{AUC} = 0.60$, Mc Neymar $p < .001$ was achieved when fitted on the test set. This suggested that the model was weakly but significantly able to classify nostalgic and non-nostalgic songs based on YouTube comments above chance-level. As such, we believe that small but significant differences exist between comments from nostalgic and non-nostalgic songs.

To understand what these features were and how they affected classification in nostalgic and non-nostalgic songs, we interpreted the model through permutation feature importance (PFI), and partial dependence plots (PDP)s by the ‘iml’ (Molnar et al., 2018) and ‘pdp’ (Greenwell, 2017) packages (all PFI scores are available in our online supplementary material: <https://osf.io/52abe/>). We instituted a cutoff of importance = 1.01 for PFI, which selected 5 features of importance for interpretation. These were Topics 4, 1, 14, and 13, as well as the JIWC-Happy emotion category. The PDPs revealed that Topics 4, 1, 14, and Happy were higher in nostalgic music comments, but Topic 13 displayed an inverted-U relationship.

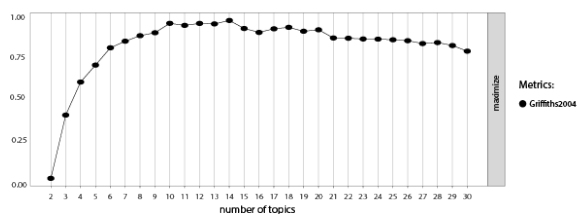


Figure 1: LDA Model likelihood at different numbers of topics for model selection.

5 Discussion

We labelled Topic 4 as ‘Bittersweet’, as it contained words that expressed both happiness and sadness, that appear self-directed and focused (e.g., ‘happiness’, ‘tears’, ‘self’, ‘believe’, ‘can-do’, and ‘find’). Topic 1 included several self-directed, high-arousal words, such as ‘live (music)’, ‘the best’, ‘favourite’, and ‘cool’, that we labelled as ‘High-arousal’. Topic 13 consisted of several words like ‘courage’, ‘sitting for entrance exams’, and ‘striving’, so we labelled it as ‘Entrance Exams’, and Topic 14 included words like ‘good’, ‘family’, ‘children’, that we labelled

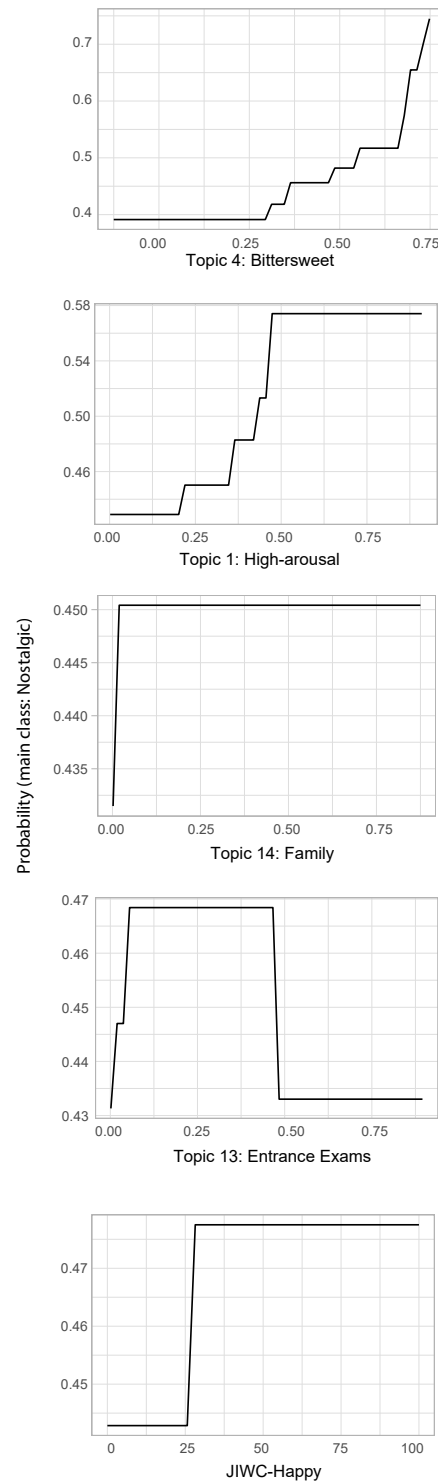


Figure 2: PDPs for high importance features (topics): the larger the probability (y-axis), the higher the probability of classification as nostalgic music. X-axis indicates the posterior probability for each topic or JIWC emotion category frequency scores

as ‘Family’. These topics, as well as happiness-related words, had an influence on the model in classifying comments. However, Topic 13 ‘Entrance Exams’ displayed a somewhat inconsistent relationship, in that the comments of low to mid probabilities on that topic were more likely to belong to nostalgic songs, but comments which were very low, and also high on that topic were from non-nostalgic songs.

Nevertheless, we conclude that comments on nostalgic songs had a greater likelihood of mentioning topics that related to bittersweet and/or high-arousal emotions, and happiness. Furthermore, they included mentions of social memories (such as family), and to a certain extent, collective memory (such as sitting for entrance exams - commonly considered a rite of passage in Japanese youth). These appear to be consistent with previously-identified appraisals and construals of Nostalgia in past literature (Sedikides and Wildschut, 2019; Tilburg et al., 2018)

However, we note the low classification accuracy of the model. It is likely that newer, more powerful models, like Latent Feature topic modeling (LFTM) and Long Short-Term Memory (LSTM) neural network classifiers, and larger sample sizes may increase the overall accuracy. Nevertheless, we believe that the consistency in interpretation with past literature adds validity to our findings, in showing for a preliminary utility in classification of emotional content of music by listener comments. This may have potential application areas such as music therapy, where ‘nostalgic’ songs can potentially be categorised efficiently and used in music-based dementia interventions (Tang et al., 2018). Our research also focused on Japanese comments for Japanese songs, but future research can extend this to different cultures and languages.

References

Marios Biskas, Wing-Yee Cheung, Jacob Juhl, Constantine Sedikides, Constantine Sedikides, Tim Wildschut, and Erica Hepper. 2019. [A prologue to nostalgia: savouring creates nostalgic memories that foster optimism](#). *Cognition and Emotion*, 33(3):417–427.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(null):993–1022.

Sergio Davalos, Altaf Merchant, Gregory Rose, Brent

Lessley, and Ankur Teredesia. 2015. ‘the good old days’: An examination of nostalgia in facebook posts. *International Journal of Human-Computer Studies*, 83:83–93.

Jerome H. Friedman. 2001. Greedy function approximation: A gradient boosting machine. *Ann. Statist.*, 29(5):1189–1232.

Brandon Greenwell, Bradley Boehmke, Jay Cunningham, and GBM Developers. 2019. *gbm: Generalized Boosted Regression Models*. R package version 2.1.5.

Brandon M. Greenwell. 2017. [pdp: An r package for constructing partial dependence plots](#). *The R Journal*, 9(1):421–436.

Thomas Griffiths and Mark Steyvers. 2004. [Finding scientific topics](#). *Proceedings of the National Academy of Sciences of the United States of America*, 101 Suppl 1:5228–35.

Bettina Grun and Kurt Hornik. 2011. [topicmodels: An R package for fitting topic models](#). *Journal of Statistical Software*, 40(13):1–30.

Motohiro Ishida. 2018. *RMeCab: interface to MeCab*. R package version 1.00.

Carol Lynne Krumhansl and Justin Adam Zupnick. 2013. [Cascading reminiscence bumps in popular music](#). *Psychological Science*, 24(10):2057–2068. PMID: 24006129.

Taku Kudo. 2005. Mecab : Yet another part-of-speech and morphological analyzer.

Max Kuhn. 2019. *caret: Classification and Regression Training*. R package version 6.0-82.

Christoph Molnar, Bernd Bischl, and Giuseppe Casalicchio. 2018. [iml: An r package for interpretable machine learning](#). *Journal of Open Source Software*, 3(26):786.

Robert Plutchik. 1980. [A general psychoevolutionary theory of emotion](#). In Robert Plutchik and Henry Kellerman, editors, *Theories of Emotion*, pages 3 – 33. Academic Press.

Lorenzo Porcaro and Horacio Saggin. 2019. [Recognizing musical entities in user-generated content](#). *Computación y Sistemas*, 23.

R Core Team. 2019. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Chelsea A. Reid, Jeffrey D. Green, Tim Wildschut, and Constantine Sedikides. 2015. [Scent-evoked nostalgia](#). *Memory*, 23(2):157–166. PMID: 24456210.

Constantine Sedikides and Tim Wildschut. 2019. [The sociality of personal and collective nostalgia](#). *European Review of Social Psychology*, 30(1):123–173.

- Daisaku Shibata, Shoko Wakamiya, Kaoru Ito, and Eiji Aramaki. 2017. *Jiwc: Kuradosooshinngu ni yoru nihongo kanjouhyougen jisho no kousaku* [jiwc: Construction of a japanese emotion-term dictionary through crowdsourcing]. In *23rd Annual Meeting of the Association for Natural Language Processing*, Tsukuba, Japan.
- Gaurav Sood. 2019. *tuber: Access YouTube from R*. R package version 0.9.8.
- Qiubi Tang, Ying Zhou, Shuixian Yang, Wong Kwok Shing Thomas, Graeme D. Smith, Zhi Yang, Lexin Yuan, and Joanne Wai yee Chung. 2018. [Effect of music intervention on apathy in nursing home residents with dementia](#). *Geriatric Nursing*, 39(4):471 – 476.
- Liila Taruffi and Stefan Koelsch. 2014. [The paradox of music-evoked sadness: An online survey](#). *PLOS ONE*, 9(10):1–17.
- Wijnand Tilburg, Martin Bruder, Tim Wildschut, Constantine Sedikides, and Anja Göritz. 2018. [An appraisal profile of nostalgia](#). *Emotion*, 19.
- Wijnand Tilburg, Tim Wildschut, and Constantine Sedikides. 2017. [Nostalgia’s place among self-relevant emotions](#). *Cognition and Emotion*, 32.
- Joseph Timoney, Brian Davis, and Adarsh Raj. 2018. [Nostalgic sentiment analysis of youtube comments for chart hits of the 20th century](#). In *AICS*.
- Matthew Vess, Jamie Arndt, Clay Routledge, Constantine Sedikides, and Tim Wildschut. 2012. [Nostalgia as a resource for the self](#). *Self and Identity*, 11.

