COMPARING RNN PARAMETERS FOR MELODIC SIMILARITY

Tian Cheng, Satoru Fukayama, Masataka Goto
National Institute of Advanced Industrial Science and Technology (AIST), Japan
{tian.cheng, s.fukayama, m.goto}@aist.go.jp

ABSTRACT

Melodic similarity is an important task in the Music Information Retrieval (MIR) domain, with promising applications including query by example, music recommendation and visualisation. Most current approaches compute the similarity between two melodic sequences by comparing their local features (distance between pitches, intervals, etc.) or by comparing the sequences after aligning them. In order to find a better feature representing global characteristics of a melody, we propose to represent the melodic sequence of each musical piece by the parameters of a generative Recurrent Neural Network (RNN) trained on its sequence. Because the trained RNN can generate the identical melodic sequence of each piece, we can expect that the RNN parameters contain the temporal information within the melody. In our experiment, we first train an RNN on all melodic sequences, and then use it as an initialisation to train an individual RNN on each melodic sequence. The similarity between two melodies is computed by using the distance between their individual RNN parameters. Experimental results showed that the proposed RNN-based similarity outperformed the baseline similarity obtained by directly comparing melodic sequences.

1. INTRODUCTION

Melodic similarity is a task to analyse the similarity between melodies, which has been used for music retrieval, recommendation, visualisation and so on. To compute the similarity, a melody is always represented by a sequence of monophonic, musical fragments/events (MIDI event, pitch, etc.). Current approaches usually compare two melodic sequences using the string edit distance [8, 9, 17], geometric measures [19] and N-Gram based measures [5, 27]. Alignment-based methods are applied when two melodic sequences are of different lengths [15, 23], or when events of two sequences are not corresponding to each other one by one [2]. Not only melodic sequence but also melody slopes on continuous melody contours are aligned for comparing melodic similarity [28]. Readers can refer to [25] for state-of-the-art melodic similarity methods. The existing methods focus on local features extracted from melodic sequences, such as distances between pitches or between subsets of melodic sequence (N-Gram). In addition alignment is needed when two melodic sequences are not comparable directly.

In order to deal with these drawbacks, we propose to train a generative Recurrent Neural Network (RNN) on a melodic sequence, and use the RNN parameters to represent the melodic sequence. The proposed feature (RNN parameters) projects a melodic sequence to a point in the parameter space, having two characteristics described as follows. Firstly, the feature is independent to the length of the input melodic sequence because every sequence is represented by its RNN parameters of the same dimension. Secondly, because the RNN can generate an identical sequence, we can expect that the RNN parameters contain the global, temporal information of the melody.

In our experiment, we first train an RNN on all melodic sequences from 80 popular songs as an initialisation. With the initialisation, RNNs are trained on individual melodic sequences. All the networks are trained in tensorflow. We compute the similarity between two melodic sequences by the Cosine similarity of their RNN parameters. The results show that the similarity based on RNN parameters outperforms the baseline similarity obtained by comparing the melodic sequences directly. To the best of our knowledge, this is the first study that uses parameters of generative RNNs for the purpose of computing melodic similarity.

2. RELATED WORK

In this section, we introduce related work on RNN-based melody generation models, and briefly introduce researches on word and sentence embedding for understanding semantic meanings in natural language processing.

2.1 RNN-based melody generation models

We discuss several state-of-the-art RNN-based melody generation models. The RNN-based generative models are usually applied with Long Short Term Memory (LSTM) units in order to model a long time dependence, such as Melody RNN in Magenta [1] and folk-rnn [22]. Magenta [1] uses 2-layer RNNs with 64 or 128 LSTM units per layer, while folk-rnn [22] uses a deeper network (RNN with 3 hidden layers of 512 LSTM units for each layer).

The RNNs generate melody by predicting the next melodic event based on its previous N events:

$$[x_{t-N}, ..., x_{t-1}] \rightarrow x_t,$$

Models	Representation	Architecture
Magenta [1]	MIDI event	2-layer RNN (LSTM)
Folk-rnn [22]	abc notation	3-layer RNN (LSTM)
Hierarchical RNN [26]	bar profile, beat profile and note	3 RNNs (2-layer LSTM) for bar, beat and note

Table 1: Brief summary of RNN-based melody generation models.

where x_t denotes the melodic event in time t. The melodic event can be represented in many forms, for example MIDI events [1], abc notation [22] and so on, as shown in Table 1. With quantised time steps (in sixteenth notes, for example), a melody can be represented as a sequence of pitches [1] or MIDI events (pitch onset, offset, and no event) [1].

Rhythm information can also be modelled for melody generation. One simple way is to concatenate beat information with the melodic event for each frame to feed into the network [1]. There are also several hierarchical RNNs proposed with rhythm information. In [4], each note is represented by its pitch and duration, and 2 RNNs (rhythm and melody RNNs) are trained for duration and pitch, respectively. The rhythm network receives the current pitch and duration as inputs, and outputs the duration of the next note. The melody network receives the current pitch and generated upcoming duration as inputs to generate the pitch of the next note. [26] trains 3 RNNs for bar, beat, and note, respectively. The first RNN generated bar profiles. Generated bar profiles are fed into the second network to generate beats, and then bar and beat profiles are fed into the third network to generate notes.

Studies of generative RNN models always list generated examples [1, 22] as results, or conduct a listening test for evaluation [26]. We believe that the generative RNN actually learns something 'musical' and can be used for music analysis. In this paper we extend the utility of the generative RNN to represent a melody and evaluate it in a melodic similarity task.

2.2 Word embedding and sentence embedding

In natural language processing, word embedding and sentence embedding work on representing semantic meanings of words and sentences. There are two successful word embedding models introduced in [13, 14]: word representations are learnt in order to predict surrounding words or to predict the current word by its content. In these ways, the meaning of a word is related to its context. With the embedded words, a representative vector for a sentence (a sequence of words) can be learned at the same time of parsing the sentence [21] or can be trained in a weakly supervised way on the click-through data by making sentence vectors with similar meanings close to each other [18]. Inspired by word embeding, [11] learns to represent a paragraph by predicting words in the paragraph using previous words and a paragraph vector. The same paragraph vector is shared when predicting words in the paragraph and then is used to represent the paragraph.

We believe that word embedding may correspond to chord embedding [3, 12] in understanding music; and sentence embedding may correspond to representing a sequence of chords (also an interesting topic to investigate). In general, the musical meaning (of a sequence of pitches or chords) is less intuitive than the textual meaning (of a word or a sentence). Thus, it is more difficult to learn a good representation for a musical sequence. In this paper we work on representing a melody (a sequence of pitches). We train an RNN model to predict the current pitch by its previous pitches in a melody and represent the melody by the RNN parameters. To the best of our knowledge, this is the first work to use network parameters directly as a representation.

3. TRAINING RNNS

For each melodic sequence, we train a generative RNN on it. The parameters of the trained RNN will be used as a feature to represent the melody. We first train an initialisation on all melodic sequences, and then train on individually melodic sequences with the initialisation.

3.1 Data

We conduct the experiment on the RWC Music Database (Popular Music) [7]. There is a subjective similarity study [10] undertaken on 80 songs (RWC-MDB-P-2001 No.1-80) of the RWC Music Database. In this study 27 participants are asked to vote the similarity (on melody, rhythm, vocals and instruments, respectively) for 200 pairs of clips after listening to them. Each clip lasts for 30 seconds (starting from the first chorus starting time). For these pairs of clips, the similarity votes range from 0 to 27. [2] The larger the vote is, the more similar the clips are. The melodic similarity matrix is shown in Figure 1, indicating the similarity scores of 200 pairs of clips. The matrix is symmetric because if a is similar to b, it means that b is similar to a as well. There are 400 non-zero values in the matrix (twice of 200 because of the symmetry).

We use the same 30-second clip as in the subjective study [10] from each song for training RNNs. We denote the clip from piece 'RWC-MDB-P-2001 No.X' as clip X, X$\in [1, 80]$. The melodic similarity results of this study [10] are used as the ground truth for evaluation.

3.2 Arranging the training data

We train RNNs using the melody annotation of the RWC Music Database (Popular Music) from the publicly available AIST Annotation [6]. A melody in the annotation is represented as a fundamental frequency sequence in 10 ms frames as shown in Figure 2(a). We call the frames with frequencies 'melody frames', and the frames without frequencies 'silent frames'. We convert the frequencies (f)

[1] `https://brangerbriz.com/blog/using-machine-learning-to-create-new-melodies/`

[2] The dataset [10] has been publicly available on the web page of the RWC Music Database at `http://staff.aist.go.jp/m.goto/RWC-MDB/AIST-Annotation/SSimRWC/`.

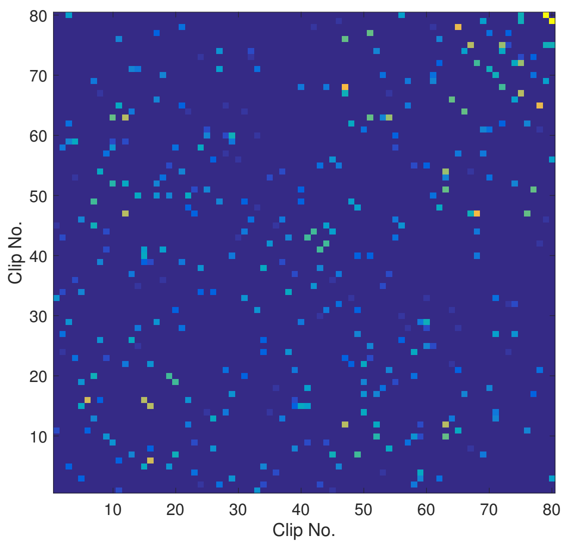


Figure 1: The melodic similarity of 200 pairs of clips.

into pitches (p – indicated by MIDI indices) for melody frames:

$$p = 69 + 12 \log_2 \frac{f}{440}. \tag{1}$$

The histogram of the pitches in the training set is shown in Figure 3. We focus on pitches in 3 octaves ranging from 43 to 78. Frames with pitches beyond this range are considered as silent frames.

3.2.1 *Frame hop size*

The original frames are arranged in a hop size of 10 ms. We use a hop size of 50 ms (shown in Figure 2(b)) because RNNs tend to repeat the previous frames with a small frame hop size.

3.2.2 *Skip silent frames*

Because of the high ratio of the silent frames (shown in Figure 2(b)), there will be many invalid training samples with a sequence of silent frames to predict a silent frame if we use all frames in the training data. Therefore, we simply skip all the silent frames to discard those invalid training samples, resulting in a pitch sequence with only melody frames (shown in Figure 5(b)).

We aim to look back for 2 seconds to predict the next frame. With a frame hop size of 50 ms, there are 40 frames in the input sequence: $[x_{t-N}, ..., x_{t-1}] \rightarrow x_t, N = 40$.

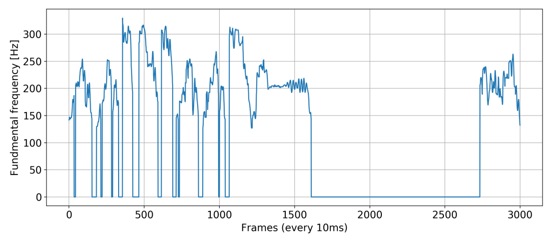
3.2.3 *Zero-padding at the beginning*

We find if the first training sample is $[x_0, ..., x_{39}] \rightarrow x_{40}$, then the generation of the first 40 frames are not modelled in the RNN. In order to generate the whole sequence, we concatenate a sequence of 40 silent frames in the front of each clip, with the first training sample of $[x_S, ..., x_S] \rightarrow x_0$ (x_S is the silent frame padding in the front of the clip).

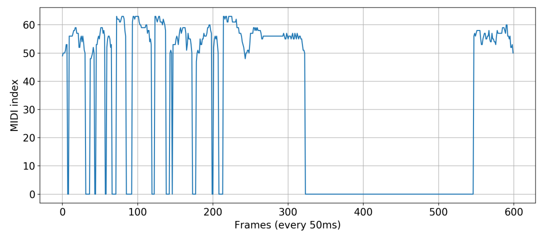
3.3 Network architecture

We apply a network architecture similar to Megenta [1], but with GRU cells instead of LSTM cells to reduce the parameter dimensions. The RNN contains 2 hidden layers with 64 GRU cells per layer. The output layer is a fully-connected layer with a softmax activation function. The inputs are one-hot encoded vectors with a dimension of 37 (36 pitches and a silent state). We hope the RNN can fit the individual pitch sequences as much as possible. In this case, overfitting is intended and not a problem any longer; hence no drop out is applied.

The network is trained by minimising the cross entropy loss using Adam optimisation with learning rate of 0.001 (other parameters of Adam are with default values in tensorflow).



(a) A sequence of fundamental frequencies.



(b) A sequence of pitches

Figure 2: Melodic sequences with different frame hop sizes. Frames with values of 0 are silent frames.

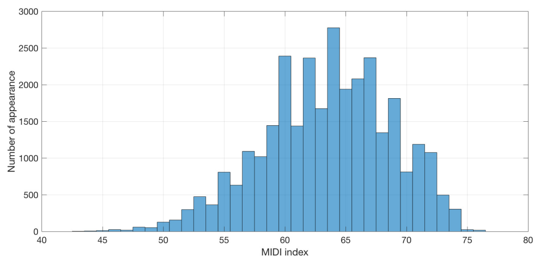


Figure 3: The histogram of the pitches in the dataset.

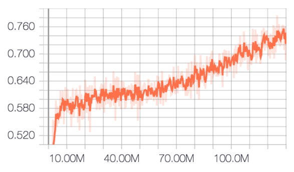
3.4 Initialisation and training on individual clips

In order to gain a consistent training, we use a fixed initialisation. The initialisation is trained on the training samples from all 80 clips for 100 epochs. Then with this initialisation, we train an individual RNN on each melodic sequence for 500 iterations.[3] After data arrangement of Section 3.2, there are around 200-600 training samples for

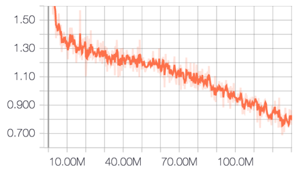
[3] An iteration means RNN parameters are updated once on a batch of training samples. In contrast, an epoch means a full training on all training samples. We use the iteration number to stop training because in this way RNN parameters are updated for the same times, hence more comparable. However, when to stop training still needs further investigation.

	Initialisation	Individual RNNs
No. of RNNs	1 RNN	80 RNNs
Training data	80 clips	each clip
Batch size	512	64
Early stop	100 epochs	500 iterations

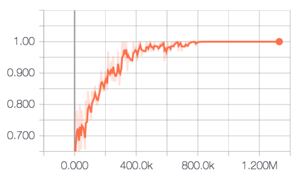
Table 2: RNN training settings.



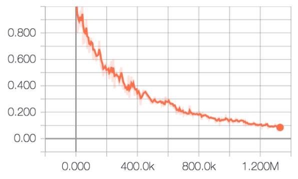
(a) Batch acc. for initialisation.



(b) Batch loss for initialisation.



(c) Batch acc. for training on clip 1.



(d) Batch loss for training on clip 1.

Figure 4: Batch accuracies and losses of training for initialisation and training on clip 1 with the initialisation.

every clip. We use a large batch size of 512 for initialisation training because of a big number of training samples, and a smaller batch size of 64 for training for each individual sequence. Training settings are shown in Table 2.

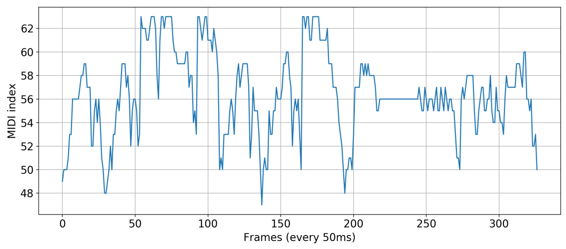
Training for initialisation and training on clip 1 are shown in Figure 4. After training for initialisation, the batch accuracy reaches 0.7 (Figure 4(a)) and the batch loss decreases to around 0.8 (Figure 4(b)). After training on clip 1 with the initialisation, the batch accuracy further increases from 0.7 to 1 (Figure 4(c)); and the batch loss reduces from 0.8 to around 0.1 (Figure 4(d)). With the RNN trained on clip 1, we can generate an identical melodic sequence, as shown in Figure 5.

3.5 Cosine similarity between RNN parameters

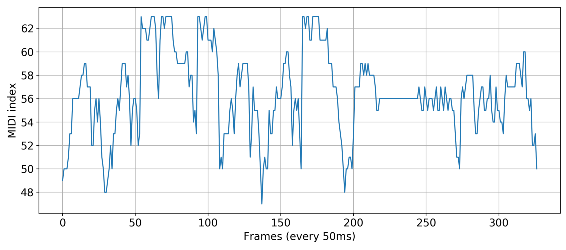
The parameter dimensions of an RNN are shown in Table 3. The total number of parameters is $46,757$.

We reshape matrices to vectors, and concatenate the vectors. The concatenated parameters of the initialisation RNN and RNNs trained on clip 3 and clip 80 are shown in Figure 6. The differences in parameters of different RNNs are subtle. The similarity between two clips is indicated by the Cosine similarity between their concatenated RNN parameters. The larger the Cosine similarity is, the more similar the clips are.

In the data arrangement stage (see Section 3.2), the melody of a clip (30 seconds) is represented as a sequence of pitches of 600 frames (including silent frames), as shown in Figure 2(b). We use the Cosine similarity between two pitch sequences as the baseline similarity.



(a) Generated pitch sequence.



(b) Original pitch sequence of clip 1.

Figure 5: An identical pitch sequence generated by the trained RNN.

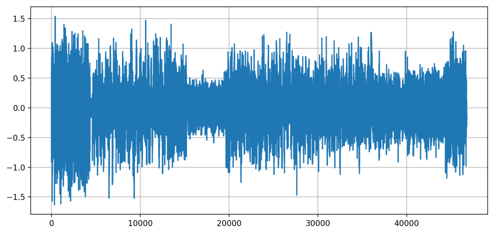
Matrix	Dimension
cell_0/gru_cell/gates/kernel	(101, 128)
cell_0/gru_cell/gates/bias	(128)
cell_0/gru_cell/candidate/kernel	(101, 64)
cell_0/gru_cell/candidate/bias	(64)
cell_1/gru_cell/gates/kernel	(128, 128)
cell_1/gru_cell/gates/bias	(128)
cell_1/gru_cell/candidate/kernel	(128, 64)
cell_1/gru_cell/candidate/bias	(64)
fully_connected/weights	(64, 37)
fully_connected/biases	(37)
all parameters	46,757

Table 3: Parameter dimensions.

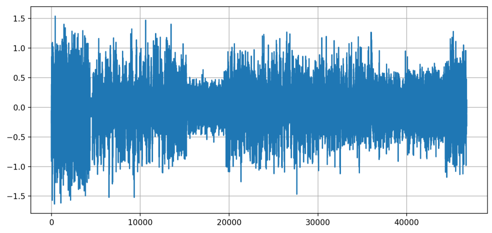
4. RESULTS ANALYSIS

4.1 Evaluation metric and results

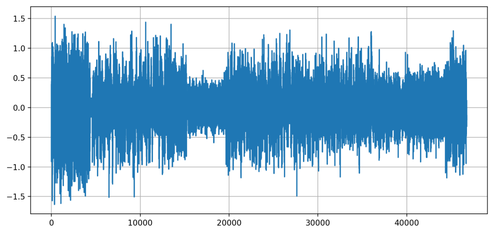
In the subjective similarity study, each clip is compared to 4-6 other clips, usually 5 clips [10]. For example, clip 3 is compared to clips as shown in Table 4(a). We measure the similarity of two clips by computing the Cosine similarity between their RNN parameters. We compare the rank of votes to the rank of similarities for evaluation. For example, as shown in Table 4(a), 8 people vote the melody of clip 80 is similar to that of clip 3, and 7 people vote the similarity between clip 29 and clip 3. Based on these votes we assume clip 80 is more similar to clip 3 than clip 29. Thus, the Cosine similarity between clip 80 and clip 3 should be larger than that between clip 29 and clip 3 $C(80,3) > C(29,3)$. We first convert the similarity and votes into ranks (as shown in Table 4(b)), and then use the pair-wise evaluation metric–Kendall's tau (τ)– to compare the ranks. For clip 3, the τ is 0.2 based on similarities between RNN parameters, better than $\tau = -0.2$ based on



(a) Parameters of the initialisation RNN



(b) Parameters of the RNN trained on clip 3



(c) Parameters of the RNN trained on clip 80

Figure 6: Parameters of different RNNs with subtle differences.

similarities between pitch sequences.

The results for 200 pairs of clips are shown in Table 5. The average τs are 0.125 and 0.073 based on Cosine similarities between RNN parameters and between pitch sequences, respectively.[4] In the preliminary test, we found that there is no improvement in performance by using a dimension-reducing technique, such as Principle Component Analysis (PCA), before computing Cosine similarity, or by using distances between eigenvectors (weighted by eigenvalues) of parameter matrices.

4.2 Visualisation

4.2.1 *Similarity v.s. vote*

We assume if there are more votes on X than on Y when comparing to A, then the X should be more similar to A than Y. However, this may be too strict when votes are close (8 on X and 7 on Y, for example). In order to show whether there is a trend that the similarity value is larger for pairs of clips with a higher vote in general, we show Cosine similarity v.s. vote plots for RNN parameters and baseline pitch sequences in Figure 7.

We know the RNN parameters of different clips are very similar to each other, as shown in Figure 6. Therefore, the

[4] Using the Euclidean distance provides similar results as using the Cosine similarity: 0.120 and 0.074 for RNN parameters and pitch sequences, respectively.

No.	80	29	59	62	5
Votes	8	7	6	4	3
C_{RNN}	0.9975	0.9973	0.99717	0.9976	0.99725
C_{pitch}	0.6175	0.7146	0.6256	0.7097	0.6584

(a) Cosine similarities between parameters of clips compared to clip 3.

No.	80	29	59	62	5	τ
R_{Votes}	1	2	3	4	5	
R_{RNN}	2	3	5	1	4	0.2
R_{pitch}	5	1	4	2	3	-0.2

(b) Ranks of Cosine similarities.

Table 4: Evaluation for clip 3. C_{RNN} and C_{pitch} are the Cosine similarities between parameters and between pitch sequences, respectively.

Similarity	τ
C_{RNN}	0.125
C_{pitch}	0.073

Table 5: Results.

Cosine similarities between RNN parameters are in a small range from 0.995 to 0.999 (Figure 7(a)). The Cosine similarities between melodic sequences are in a larger range from 0.4 to 0.9 (Figure 7(b)). However, neither RNN parameters nor melodic sequences provide a clear trend of the similarity increasing with number of votes.

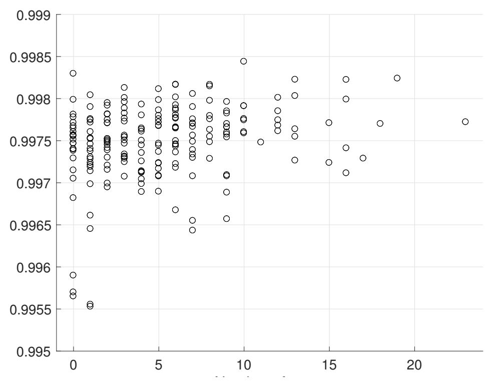
4.2.2 *t-SNE*

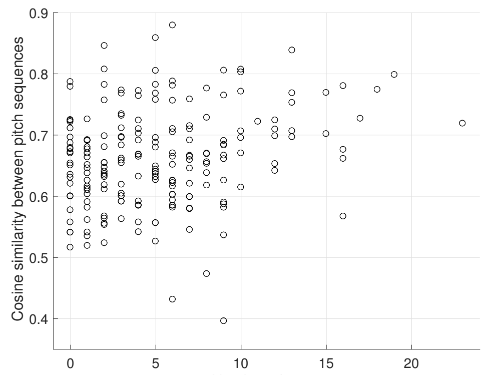
To visualise the 80 songs in a low-dimensional space, we first reduce the dimension of the features to 5 by PCA, then further reduce it to 2 by t-SNE, with the implementation of [20]. The visualisation based on RNN parameters and pitch sequences is shown in Figure 8. For a clearer visualisation, we only indicate pairs of clips with higher votes (above 9 votes out of 27, as listed in Table 6) by connecting those pairs with lines.

Because the t-SNE visualisation is not a linear projection from the similarity to the distance on the 2-dimensional space, we do not compare the vote against the distance between two clips in t-SNE visualisation, but focus on the grouping of clips. We observe some interesting grouping of clips in Figure 8(a): the triangle at the top left for (75, 79, 80), and two lines at bottom right connecting (15, 16) and (6,16). In Figure 8(b), no such grouping of clips can be obviously observed.

5. DISCUSSIONS AND CONCLUSIONS

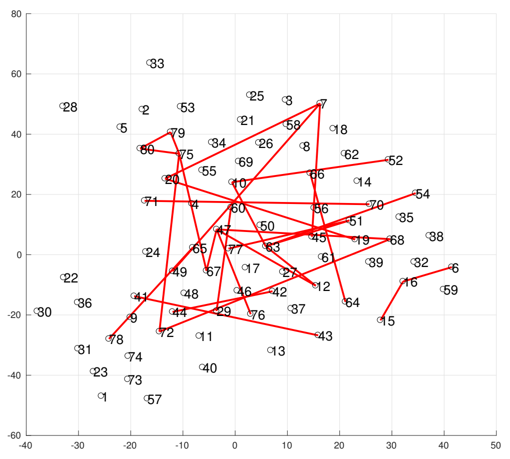
From the t-SNE visualisation, we observe some interesting grouping of clips based on RNN parameters (Figure 8(a)). However, visualisation based on the Cosine similarity between RNN parameters does not show a clear relation between the similarity and the vote (Figure 7(a)). It may


(a) Visualisation based on RNN parameters



(b) Visualisation based on pitch sequences

Figure 7: Similarity v.s. vote plot based on different features.


(a) Visualisation based on RNN parameters

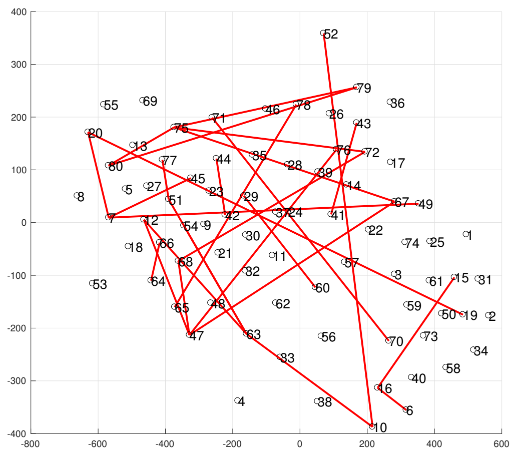

(b) Visualisation based on pitch sequences

Figure 8: t-SNE visualisation based on different features.

indicate that a direct comparison between RNN parameters is too simple to infer the information in such a large dimension. Figure 6 also illustrates the difficulties with the proposed approach, too many parameters with subtle differences. We would like to dig deeper to understand which parameters are most significant for computing melodic similarity.

Perception studies show that changes in relative scale or relative duration do not have a major impact on melodic similarity [24]. The similarity measure should be invariant to music transformations, such as transposition in pitch and tempo changes [16,23]. The proposed generative RNN can model the input pitch sequence, but cannot deal with the similarity under music transformations. In the future, we would like to tackle this problem by training RNNs with coordinate differences instead of absolute coordinates as inputs, such as intervals and durations instead of pitches and onsets [16].

We work on the melodic similarity based on the performance-based representation of melodies, which seems to complicate the task. We hope we can achieve more success on symbolic melody representation by using score-based representation on a simpler dataset.

In this paper, we propose to represent a melodic sequence by the parameters of its corresponding generative RNN, and test the utility of the melodic feature (RNN parameters) in the melodic similarity task. The proposed feature contains temporal information within the melodic sequence, and independent of the length of the sequence. We extend the utility of generative RNNs to use the network for music similarity analysis rather than music generation. We expect that the proposed feature (generative RNN parameters) can be used in other tasks, such as musicological analysis and music cognition.

No.	Pair	Vote	No.	Pair	Vote	No.	Pair	Vote
1	(79, 80)	23	11	(10, 63)	13	21	(10, 52)	11
2	(47, 68)	19	12	(47, 76)	13	22	(7, 20)	10
3	(65, 78)	18	13	(51, 63)	13	23	(7, 45)	10
4	(6, 16)	17	14	(51, 77)	13	24	(29, 60)	10
5	(12, 47)	16	15	(64, 66)	13	25	(47, 67)	10
6	(12, 63)	16	16	(7, 49)	12	26	(70, 71)	10
7	(15, 16)	16	17	(19, 20)	12	27	(75, 79)	10
8	(67, 75)	16	18	(41, 43)	12	28	(75, 80)	10
9	(54, 63)	15	19	(42, 44)	12			
10	(72, 75)	15	20	(68, 72)	12			

Table 6: A list of pairs of songs with similarity votes above 9 votes out of 27.

6. ACKNOWLEDGEMENT

This work was supported in part by JST ACCEL Grant Number JPMJAC1602, Japan.

7. REFERENCES

[1] Magenta: Melody RNN. `https://github.com/tensorflow/magenta/tree/master/magenta/models/melody_rnn`. Accessed: 2018-03-30.

[2] D. Bountouridis, D. G. Brown, F. Wiering, and R. C. Veltkamp. Melodic Similarity and Applications Using Biologically-Inspired Techniques. *Applied Sciences*, 7(12), 2017.

[3] G. Brunner, Y. Wang, R. Wattenhofer, and J. Wiesendanger. JamBot: Music Theory Aware Chord Based Generation of Polyphonic Music with LSTMs. In *Proceedings of the 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, 2017.

[4] F. Colombo, S. P. Muscinelli, A. Seeholzer, J. Brea, and W. Gerstner. Algorithmic Composition of Melodies with Deep Recurrent Neural Networks. *Computing Research Repository (CoRR)*, abs/1606.07251, 2016.

[5] J. S. Downie. *Evaluating a Simple Approach to Musical Information retrieval: Conceiving Melodic N-grams as Text*. PhD thesis, University of Western Ontario, 1999.

[6] M. Goto. AIST Annotation for the RWC Music Database. In *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, pages 359–360, 2006.

[7] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC Music Database: Popular, Classical, and Jazz Music Databases. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, pages 287–288, 2002.

[8] M. Grachten, J.-L. Arcos, and R. L. de Mantaras. Melodic Similarity: Looking for a Good Abstraction Level. In *Proceedings of the 5th International Society of Music Information Retrievall (ISMIR)*, 2004.

[9] P. Hanna, P. Ferraro, and M. Robine. On Optimizing the Editing Algorithms for Evaluating Similarity Between Monophonic Musical Sequences. *Journal of New Music Research*, 36(4):267–279, 2007.

[10] S. Kawabuchi, C. Miyajima, N. Kitaoka, and K. Takeda. Subjective Similarity of Music: Data Collection for Individuality Analysis. In *Proceedings of Asia Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 1–5, 2012.

[11] Q. Le and T. Mikolov. Distributed Representations of Sentences and Documents. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2014.

[12] S. Madjiheurem, L. Qu, and C. Walder. Chord2Vec: Learning Musical Chord Embeddings. In *Proceedings of the Constructive Machine Learning Workshop at 30th Conference on Neural Information Processing Systems (NIPS)*, 2016.

[13] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean. Efficient Estimation of Word Representations in Vector Space. In *Proceedings of International Conference on Learning Representations (ICLR) Workshop*, 2013.

[14] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Proceedings of Advances in Neural Information Processing Systems 26 (NIPS)*, pages 3111–3119, 2013.

[15] D. Müllensiefen and K. Frieler. Optimizing Measures of Melodic Similarity for the Exploration of a Large Folk Song Database. In *Proceedings of the 5th International Society of Music Information Retrievall (ISMIR)*, pages 1–7, 2004.

[16] D. Müllensiefen and K. Frieler. Evaluating Different Approaches to Measuring the Similarity of Melodies. In et al. V. Batagelj, editor, *Data Science and Classification*, pages 299–306. Springer, Berlin, 2006.

[17] K. S. Orpen and D. Huron. Measurement of Similarity in Music: A Quantitative Approach for Non-parametric Representations. *Computers in Music Research*, 4:1 – 44, 1992.

[18] H. Palangi, P. li, Y. Shen, J. Gao, X. He, J. Chen, X. Song, and R. Ward. Deep Sentence Embedding Using Long Short-Term Memory Networks: Analysis and Application to Information Retrieval. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 24(4):694–707, 2016.

[19] M. W. Park and E. C. Lee. Similarity Measurement Method between Two Songs by Using the Conditional Euclidean Distance. *Wseas Transaction On Information Science And Applications*, 10(12), 2013.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-Learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[21] R. Socher, C. Y. Lin, A. Y. Ng, and C. D. Manning. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *Proceedings of International Conference on Machine Learning (ICML)*, 2011.

[22] B. L. Sturm, J. F. Santos, and I. Korshunova. Folk Music Style Modelling by Recurrent Neural Networks with Long Short Term Memory Units. In *Extended abstracts for the Late-Breaking Demo Session of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, 2015.

[23] J. Urbano, J. Lloréns, J. Morato, and S. Sánchez-Cuadrado. MIREX 2012 Symbolic Melodic Similarity: Hybrid Sequence Alignment with Geometric Representations. In *Music Information Retrieval Evaluation eXchange (MIREX)*, 2012.

[24] M. R. Velankar, H. V. Sahasrabuddhe, and P. A. Kulkarni. Modeling Melody Similarity Using Music Synthesis and Perception. *Procedia Computer Science*, 45:728 – 735, 2015.

[25] V. Velardo, M. Vallati, and S. Jan. Symbolic Melodic Similarity: State of the Art and Future Challenges. *Computer Music Journal*, 40(2):70–83, 2016.

[26] J. Wu, C. Hu, Y. Wang, X. Hu, and J. Zhu. A Hierarchical Recurrent Neural Network for Symbolic Melody Generation. *Computing Research Repository (CoRR)*, abs/1712.05274, 2017.

[27] S. Yazawa, Y. Hasegawa, K. Kanamori, and M. Hamanaka. Melodic Similarity Based on Extension Implication-Realization Model. In *Music Information Retrieval Evaluation eXchange (MIREX)*, 2013.

[28] Y. Zhu, M. Kankanhalli, and Q. Tian. Similarity Matching of Continuous Melody Contours for Humming Querying of Melody Databases. In *Proceedings of IEEE Workshop on Multimedia Signal Processing*, 2002.