HOW MUSIC FEATURES AND MUSICAL DATA REPRESENTATIONS AFFECT OBJECTIVE EVALUATION OF MUSIC COMPOSITION: A REVIEW OF THE CSMT DATA CHALLENGE 2020

Yuqiang Li[1] **Shengchen Li**[1] **George Fazekas**[2]
[1]Xi'an Jiaotong-Liverpool University
[2]Queen Mary University of London
{yuqiang.li19@student., shengchen.li@}xjtlu.edu.cn,
g.fazekas@qmul.ac.uk

ABSTRACT

Tools and methodologies for distinguishing computer-generated melodies from human-composed melodies have a broad range of applications from detecting copyright infringement through the evaluation of generative music systems to facilitating transparent and explainable AI. This paper reviews a data challenge on distinguishing computer-generated melodies from human-composed melodies held in association with the Conference on Sound and Music Technology (CSMT) in 2020. An investigation of the submitted systems and the results are presented first. Besides the structure of the proposed models, the paper investigates two important factors that were identified as contributors to good model performance: the specific music features and the music representation used. Through an analysis of the submissions, important melody-related music features have been identified. Encoding or representation of the music in the context of neural network modes are found noticeably impacting system performance through an experiment where the top-ranked system was re-implemented with different input representations for comparison purposes. Besides demonstrating the feasibility of developing an objective music composition evaluation system, the investigation presented in this paper also reveals some important limitations of current music composition systems opening opportunities for future work in the community.

1. INTRODUCTION

With the rapid development of AI, automatic music composition systems are considered to approach the quality of human-composed melodies in their output under certain conditions. The Conference on Sound and Music Technology (CSMT) organised a data challenge in 2020 to investigate how to tell apart human-composed melodies from computer-generated melodies, where participants were required to develop an algorithm (or a system) that can distinguish computer-generated melodies from human-composed ones. The submitted algorithms can potentially be used to prevent computer systems from being accused of music copyright infringement, and to objectively measure the performance of different algorithmic composition systems.

The primary task in this data challenge was to develop a system to identify human-composed melodies in a dataset where they are mixed with computer-generated melodies. The generated melodies come from several state-of-the-art music generation frameworks [1], including Variational Auto-Encoder (VAE) [2], Transformer [3] and Generative Adversarial Network (GAN) [4]. A training set with only generated melodies was released first, followed by an evaluation set with human-composed melodies mixed in. Two datasets were used for training the music generation systems separately: Bach Chorales [1] in Music21 [5] and HookTheory [2]. The generated and composed melodies thus followed two styles associated with Bach and more generally the pop genre.

The rankings of the participants were determined by the AUC score in ROC tests, based on the task of identifying human-composed melodies. 7 teams participated in the data challenge with a total of 14 submitted systems, covering various types of algorithms. For instance, rule-based system [6], LSTM (Long short-term memory) [7–9], AE (auto-encoder) [10] and more conventional SVM (support vector machine) [11]. The top-ranked submission by Li *et. al* [7] obtained an AUC score of 0.88, which demonstrated the feasibility of using a computer to identify the music source, at least in a specific context of the two styles by distinguishing human compositions from generated melodies.

The results of the data challenge are further investigated in three ways: style, pitch feature and music representation. The sub-rankings regarding the two music styles are also compared, though little difference was observed. However, the system performance shows significant differences in terms of different pitch features in melodies and music representation methods. Moreover, the paper also presents a revised version of the top-ranked system with

© Y. Li, S. Li, and G. Fazekas. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Y. Li, S. Li, and G. Fazekas, "How Music Features and Musical Data Representations Affect Objective Evaluation of Music Composition: A Review of the CSMT Data Challenge 2020", in *Proc. of the 23rd Int. Society for Music Information Retrieval Conf.,* Bengaluru, India, 2022.

[1] https://web.mit.edu/music21/doc/moduleReference/moduleCorpusChorales.html
[2] https://www.hooktheory.com/

different input representations to discuss how music representation affects the identification of melodies.

The identification of computer-generated melodies could be used to ensure that no copyright is claimed for fully-automated computer-generated melodies. [12]. Moreover, measuring similarity between human and computer-generated melodies could become a component of Generative Adversarial Networks (GAN) and similar frameworks for automatic and assistive composition, as well as reduce the subjective bias in the evaluation of music generation systems.

The remainder of the paper is organised as follows: we first review the CSMT 2020 data challenge including submissions and results. The performance differences between submissions regarding different musical features are then investigated. Two experiments on the impacts of music representations are presented followed by a brief conclusion.

2. DATA CHALLENGE

2.1 Task Definition and Organisation

The proposed task for the data challenge was to distinguish human-composed melodies from computer-generated melodies, all in the form of 8-bar single-track MIDI files. A training dataset was released first, containing only computer-generated melodies. The evaluation dataset, with equal numbers of human-composed melodies and generated melodies, was released without the associated labels one month before the final submission deadline. The final results were ranked using the AUC score in ROC test, which identifies human-composed melodies from computer-generated ones.

Time	Event
July 15th, 2020	Release of the development dataset
August 15th, 2020	Release of the evaluation dataset
September 15th, 2020	Deadline of submission (prediction, model and report)
October 20th, 2020	Submission review finished
November 4th, 2020	Result announcement

Table 1: Timeline of the CSMT 2020 Data Challenge

2.2 Dataset

A detailed specification of the dataset, especially the training dataset, can be found in [1]. The components of evaluation set are shown in Table 2 to provide a clearer context of the results presented in this paper. Notice that according to [1], 95% of the human-composed melodies in the evaluation dataset are randomly sampled from those that were used to train the three types of generative models.

2.3 Baseline System

An Auto-Encoder (AE) neural network was used as the baseline system of the challenge, whose source code is available on the official website. Figure 1 plots the model architecture. The model uses a bottleneck structure with the cross entropy loss.

	MTrans	MVAE	MNet	Human	Total
Bach	600	200	200	1000	2000
Pop	600	200	200	1000	2000
Total	1200	400	400	2000	4000

Table 2: The evaluation dataset used in the challenge. “MTrans”, “MVAE”, “MNet” are short for Music Transformer [13], MusicVAE [2] and MidiNet [4], respectively. Music styles “Bach” and “Pop” are listed separately.

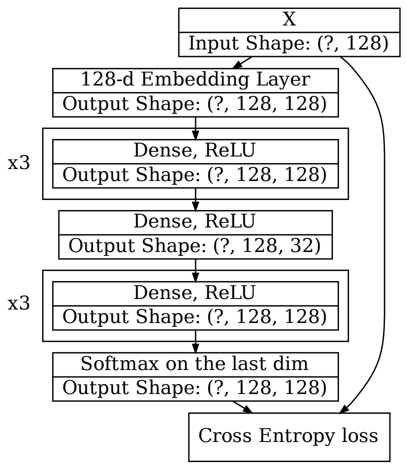
X
Input Shape: (?, 128)
128-d Embedding Layer
Output Shape: (?, 128, 128)
x3
Dense, ReLU
Output Shape: (?, 128, 128)
Dense, ReLU
Output Shape: (?, 128, 32)
x3
Dense, ReLU
Output Shape: (?, 128, 128)
Softmax on the last dim
Output Shape: (?, 128, 128)
Cross Entropy loss

Figure 1: Baseline Auto-Encoder Network Architecture

The baseline model was trained with only real melodies from the Nottingham Folk Tunes dataset[3], consisting of 1,200 American and British songs. The chord information provided in the dataset was not used. The baseline system characterises features of human-composed melodies with an auto-encoder, where computer-generated melodies will result in a high reconstruction loss.

2.4 Overview of the Submitted Systems

For each submitted system, participants were allowed to provide multiple predictions based on different settings or parameters of a model. The challenge received 14 sets of predictions submitted by 7 participants. To simplify references to the submitted systems, aliases are introduced in Table 3 to represent the 8 models. In the table, the presented AUC score is the highest among all predictions of each submitted system.

Around half of the submitted systems outperformed the baseline, these are listed in Table 3. Most participants adopted Deep Neural Networks (DNN), but conventional ML algorithms and rule-based methods were also observed.

There are three main types of methodologies among the submissions: outlier detection, binary classification and heuristics. The outlier detection systems [10] and [14] only learn the feature distribution of the samples of one class

[3] `https://github.com/jukedeck/nottingham-dataset`

and considers the other class as outliers. Binary classifiers [7–9, 11] use both positive and negative samples for training. The rule-based system [6] uses a set of heuristic rules to test the input melody.

The outlier detection method adopts the idea of Task 2 from the DCASE data challenge 2020. With this configuration, an auto-encoder system is trained in an unsupervised manner for either computer-generated melodies or human compositions. The samples from the outlier class are expected to result in a high reconstruction loss. Besides the baseline system, Ding & Ma [10] and Yu *et al.* [14] used this idea and were ranked 4th and 8th respectively.

The best-performing system [7], Guo *et al.* [8], Xia *et al.* [9], and Wang *et al.* [11] adopted the idea of binary classification. External data, such as Wikifonia, Nottingham and Midi Man [4] . was used to train the classifiers with both positive and negative samples.

The only rule-based system [6] has ranked 2nd as a team, and 3rd as a system, outperforming quite a few ML and DNN systems. In summary, this system takes an 8-bar melody as input and finds the most possible key centers for each bar. The unusual key changes are then counted for every 2 consecutive bars. The key center of a bar is determined by finding out a natural major or harmonic minor scale among all possible scales such that this scale contains most of the notes in this bar. Unusual key changes are defined as any modulations to a distant key, which means the tonic of the new key is 3 fifths or more apart from the original tonic. The success of the system points out that there is still a long way to go for computer music generation systems in learning particular music theories.

3. IMPACT OF STYLE

The evaluation dataset contains equal number of Bach-style and Pop-style melodies. The model performances in each of these two subsets are listed as AUC scores in the column *Bach* and *Pop* of Table 3. Compared with the overall AUC scores, there is no strong difference shown in the model performances given the different style subsets. An exception is that the top-ranked and the second-ranked systems by Li *et. al* are clearly better at evaluating Bach-style melodies than Pop-style ones with differences in the AUC score more than 0.1.

4. IMPACT OF PITCH FEATURES

4.1 Pitch Features

Pitch features are commonly used for the evaluation of generative music systems [15], specifically on melodies. Three pitch features of melodies are proposed for investigating how pitch features impact the performance of the submitted systems: Interval Mean (IM), Interval Standard Deviation (ISD) and Major-scale-rate Standard Deviation (MSD).

Suppose $\mathbf{X}$ is a melody denoted by a MIDI pitch sequence without duration information $[p_1, p_2, \ldots, p_n], 0 \leq p \leq 127$, the first-order forward difference of the pitch sequence $\mathbf{X}$ can be represented as $\Delta\mathbf{X} = [p_2 - p_1, p_3 - p_2, \ldots, p_n - p_{n-1}]$, each of which is an signed integer.

IM and ISD are defined as

$$\mathrm{IM}(\mathbf{X}) := \mathbb{E}[\Delta\mathbf{X}] \tag{1}$$

$$\mathrm{ISD}(\mathbf{X}) := \mathrm{Std}\left(\Delta\mathbf{X}\right) \tag{2}$$

where “Std” is the standard deviation.

Major Scale Rate $\mathrm{MSR}(\mathbf{X})$ is defined as a function that maps a melody $\mathbf{X}$ to a 12-dimensional vector $[c_0, c_1, \ldots, c_{11}]$, where c_i is the number of notes in $\mathbf{X}$ that belong to the natural major scale with pitch class i as the tonic (namely, C major, C♯ major, …, B major). The vector is normalised by dividing all components by their sum so that $\sum_i \mathrm{MSR}(\mathbf{X})_i = 1$. Major-scale-rate Standard Deviation (MSD) is obtained by

$$\mathrm{MSD}(\mathbf{X}) := \mathrm{Std}\left(\mathrm{MSR}(\mathbf{X})\right). \tag{3}$$

IM approximates the overall tendency of a pitch sequence. For example, if $\mathrm{IM}(\mathbf{X}) > 0$, the melody $\mathbf{X}$ must end at a pitch higher than the starting pitch [5] . ISD measures the unevenness of a melody. A melody with a higher ISD usually sounds more stochastic and chaotic. MSD measures the extent to which a melody is concentrated on a specific (natural major) scale.

4.2 Impact of Pitch Features on Model Performance

For each proposed feature, the following procedure is used to compare the performance of the submitted models given melodies with different IM, ISD and MSD.

For each pitch feature, the feature values are calculated for all melodies in the evaluation dataset. According to feature values, melodies are grouped into 30 bins that equally divide the whole range of feature values. In each bin, the mean absolute error (MAE) between the predicted score and the label of the melody (0 for generated and 1 for human-composed) was used as the measurement of system performance.

4.2.1 Interval Mean

Figure 2a compares how the MAEs were distributed on different IM ranges for different submissions. A dashed KDE plot was also used to show IM distribution for all melodies in the dataset. The majority of the dataset has a IM value in the range of $[-0.75, 0.75]$.

Among all the models, as plotted in the Figure 2a, the two LSTM-based models submitted by Li *et al.* (the blue and the orange curves) produced comparatively lower errors than the others. Xia (the pink curve) submitted another LSTM-based model, with a similar performance to

[4] `https://www.reddit.com/r/datasets/comments/3akhxy/the_largest_midi_collection_on_the_internet/`

[5] Numerically, IM equals the starting pitch subtracted from the ending pitch.

Rank	Submission Name	Alias	Model	Dataset	Flags	Bach	Pop	All
1	You_Li_NYU_Zhuowen_Lin_GATech_uni [7]	Li-uni	LSTM	Midi Man (1,000), CSMT (6,000) [1]	DCA	0.94	0.83	0.88
2	You_Li_NYU_Zhuowen_Lin_GATech_bi [7]	Li-bi	Bi-LSTM			0.89	0.71	0.80
3	Yang_Deng_Netease [6]	Deng	Statistics	(None)	S..	0.77	0.76	0.76
4	Mingshuo_Ding_PKU_Yinghao_Ma_CMU [10]	Ding	ALBERT AE	Fake only, CSMT (6,000)	DRA	0.70	0.67	0.68
5	Jingyue_Guo_PATech [8]	Guo	Bi-LSTM	Wikifonia (6,675), CSMT (6,000)	DC.	0.61	0.65	0.63
-	Baseline	Baseline	MLP-AE	Real only, Nottingham (1,034)	DR.	0.58	0.63	0.61
6	Yiting_Xia [9]	Xia	LSTM	Real* (5,742), CSMT (6,000)	DCA	0.57	0.64	0.60
7	Jiaxing_Yu_ZJU [14]	Yu	CNN-AE	Fake only, MusicVAE (4,000)	DR.	0.62	0.48	0.56
8	Yuxiang_Wang_BIT [11]	Wang	SVM	Nottingham (1,034), CSMT (6,000)	MC.	0.55	0.47	0.51

*: From multiple data sources.
Flag 1: **D**: DNN, **M**: Traditional ML, **S**: Statistical. Flag 2: **C**: Classifier-based, **R**: Reconstruction-based. Flag 3: **A**: Dataset augmentation was performed.

Table 3: A list of all submissions with AUC scores for the Bach subset, Pop subset and the entire set presented.

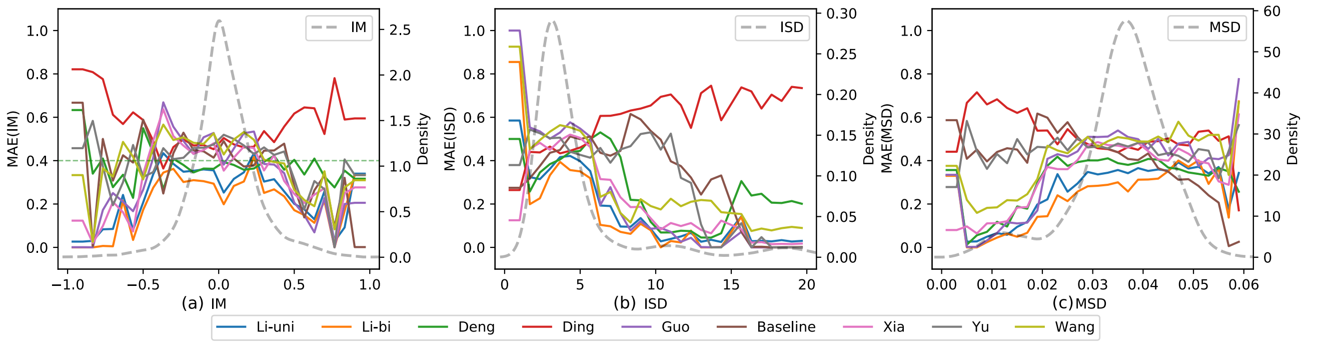

Figure 2: Model MAE on different features. The feature KDE is plotted in a dashed line style to characterise the distribution of music features.

the previous two when IM > 0.5, but higher errors when IM < 0.5. Deng's system (the green curve) makes prediction by counting the bars that modulate to a distant key. As a visual aid, a horizontal dashed green line was plotted to show that the MAE of Deng's model stays around 40%. Especially for IM in $[-0.5, 0.5]$, Deng's model surpassed a few DNN models with low errors only second to Li's models. This model also displayed a stable and consistent performance even with the melodies that have extreme IM values.

4.2.2 *Interval Standard Deviation*

ISD roughly measures the evenness of the melody. Melodies with low ISD are extremely stable and repetitive thus harder to distinguish. Melodies with high ISD tend to be stochastic and more random, which are relatively simpler to distinguish.

Figure 2b illustrates the MAE distributions of different models in different ISD bins. The leftmost highlighted area indicates that most DNN-based models (LSTM, CNN and AE) have poor performance when ISD is within $(0, 2.5]$. However, on the right hand side of Figure 2b, when it comes to melodies with ISD in $[15, 20]$ (which is extremely high), the system performance improves. This evidence shows that most models have successfully learned to recognise some melodies as the outliers if they are stochastic and contain many jumping intervals.

For melodies with higher ISD (ISD > 5), where neighbour pitches are less likely to be fully covered by the convolutional kernel, Yu's CNN-based model (grey curve) produced noticeably more errors than the rest of the models. Ding's BERT-based model (red curve) has an even higher error when ISD is high, indicating that the model is possibly over-fitted with poor generalisation.

4.2.3 *Major-scale-rate Standard Deviation*

A melody with a higher MSD tends to be more diatonic and less chromatic. Figure 2c reveals the performance of submissions regarding different MSD ranges. With lower MSD value (MSD < 0.02), all the LSTM models [7–9] retained relatively lower MAEs than the others, hinting some potential relationship between recognising atonal melodies and the characteristics of LSTM network itself.

Deng's proposed rule-based system, designated to inspecting the tonal properties of a melody also showed a low-error curve, only behind the top-ranked LSTM systems. The stability and strong generalisation property of Deng's system enabled its survival of the rare test cases. For instance, around MSD $= 0.06$, (very high, the melody being too diatonic), some ML or DNN models suddenly collapsed with high errors while Deng's system only had a slight increment in error, with the absolute value remaining less than 0.4.

IM, ISD and MSD can effectively reveal the melody characteristics and hence can be used to explain the model performance using different aspects. The distributions of these features can guide the training process of a music generation system, or as a series of a-priori weights to refine the predictions in a discriminative model according to how common the melody is.

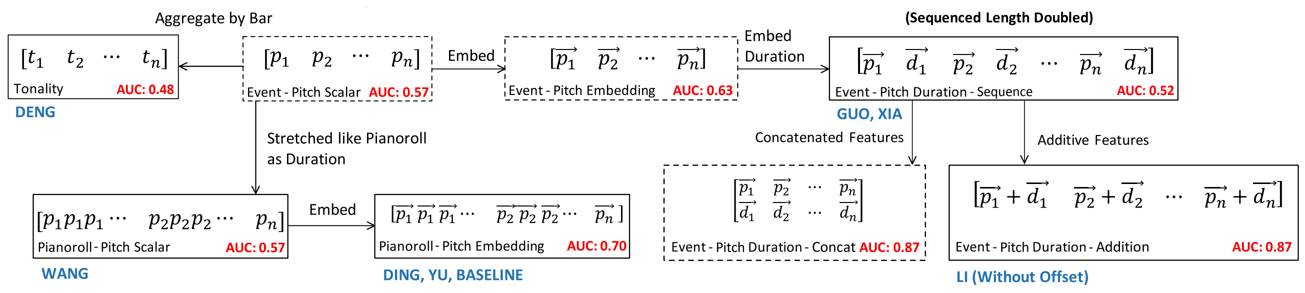
Aggregate by Bar
$[t_1 \quad t_2 \quad \cdots \quad t_n]$
Tonality AUC: 0.48
DENG
$[p_1 \quad p_2 \quad \cdots \quad p_n]$
Event - Pitch Scalar AUC: 0.57
Embed
$[\vec{p_1} \quad \vec{p_2} \quad \cdots \quad \vec{p_n}]$
Event - Pitch Embedding AUC: 0.63
Embed Duration
(Sequenced Length Doubled)
$[\vec{p_1} \quad \vec{d_1} \quad \vec{p_2} \quad \vec{d_2} \quad \cdots \quad \vec{p_n} \quad \vec{d_n}]$
Event - Pitch Duration - Sequence AUC: 0.52
GUO, XIA
Stretched like Pianoroll as Duration
Concatenated Features
Additive Features
$[p_1 p_1 p_1 \cdots \quad p_2 p_2 p_2 \cdots \quad p_n]$
Pianoroll - Pitch Scalar AUC: 0.57
WANG
Embed
$[\vec{p_1}\vec{p_1}\vec{p_1}\cdots \quad \vec{p_2}\vec{p_2}\vec{p_2}\cdots \quad \vec{p_n}]$
Pianoroll - Pitch Embedding AUC: 0.70
DING, YU, BASELINE
Event - Pitch Duration - Concat AUC: 0.87
$[\vec{p_1}+\vec{d_1} \quad \vec{p_2}+\vec{d_2} \quad \cdots \quad \vec{p_n}+\vec{d_n}]$
Event - Pitch Duration - Addition AUC: 0.87
LI (Without Offset)

Figure 3: Representation relationship diagram. The edge label denotes the transformation from a representation to another. The aliases of the participants who used a certain type of representation are listed on the left beneath the box, uppercase bold. The AUC score in the bottom right corner refers to the results of the experiment mentioned in Section 5.1.

5. IMPACT OF MUSIC REPRESENTATION

From the submissions, the differences in music representation are focused on two aspects: the input representation and the metric unit of time. Various music representations can be distinguished mainly by what features of notes are selected and how they are organised in structural data. The metric unit of time refers to the minimal time unit to denote the music data. In this study, the top-ranked system by Li *et al.* [7], an LSTM-based binary classifier, has been slightly revised to adopt different types of music representations in order to investigate the impact of music representation on the same task. All re-implemented models are trained 50 epochs using the same training dataset used by Li, (as listed in Table 3) and are evaluated on the evaluation dataset. ROC AUC is used as the evaluation strategy.

5.1 Melody Representations

Three aspects of a melody representation will be discussed in section 5.1: the evenness along time steps, the subsets of features regarding notes and the method of feature fusion.

Evenness of time steps (ETS) The term *Pianoroll* is used if the duration is implicitly represented in the form of pianoroll. One step in these always stands for a constant length of time. Otherwise, the term *event* will be used.

Subset of note features (SNF) Features regarding notes include pitch, duration, velocity [6], *etc*. Particularly in this challenge, only "*Pitch*" and "*Duration*" are used.

Feature fusion method (FFM) Feature fusion is usually achieved in the following three ways if more than one note features is used:

1. Use separate events in consecutive time steps (sequentially), *e.g.*, $\left[\vec{p_1}, \vec{d_1}, \vec{p_2}, \vec{d_2}, \cdots\right]$
2. Concatenate the embedded vectors of different features of the same note into a higher-dimensional vector, *e.g.*, $\left[\vec{p_1}\|\vec{d_1}, \vec{p_2}\|\vec{d_2}, \cdots\right]$
3. Add the embedded vectors of different features of the same note, *e.g.*, $\left[\vec{p_1}+\vec{d_1}, \vec{p_2}+\vec{d_2}, \cdots\right]$

where $\vec{p_i}$ and $\vec{d_i}$ are the embedded vectors (in the same dimension) of the pitch and duration event token for the i-th note in the melody. The representation methods will be named in a format of *ETS-SNF-FFM* as shown in Figure 3.

The only exception of representation is *Tonality* representation, as used by Deng *et. al*. Deng's system represents the input data via the tonality distributions of all bars in a melody [6]. The tonality distribution, denoted by the set of all possible key candidates, summarises the tonal characteristics at a bar level rather than a single note. This leads to the representation name *Tonality* representation.

All the melody representations used by the submitted models can be categorised into 5 types according to the three aspects, plus the extra *tonality*. Besides these that have been used, another three representations were added to better illustrate the relationship among different representations, namely *Event-Pitch-Scalar*, *Event-Pitch-Embedding* and *Event-PitchDuration-Concat*. The former two contain only pitch information, differentiated by whether embedding is used. The *Event-PitchDuration-Concat* representation concatenates pitch vectors and duration vectors instead of element-wise adding. In Li's model [7], the embedding dimension is chosen to be 128.

Figure 3 reveals how a melody representation is related to another through a certain transformation. For instance, starting from the most simple pitch-only *Event-Pitch-Scalar* representation, grouping the pitches by bar gives the *Tonality* representation. Stretching the sequence according to the note position and duration will yield a pianoroll. Using embedding will produce *Event-Pitch-Embedding* and *Pianoroll-Pitch-Embedding*.

5.2 Metric Unit of Time in Music Representation

The metric unit of time in a representation refers to the minimal time unit that the positions and durations of all notes in a melody are always the multiples of. This concept is also known as midi resolution. Quantisation is the procedure that re-samples a midi file to a specific resolution. 3 submissions chose a specific metric unit of time for pre-processing all the melodies before the model input. The results suggest that the unit of metric should be carefully determined according to the specific model. Generally, a larger metric unit will result in fewer rhythmic patterns and time steps, especially for pianoroll representations. This reduces the difficulty in modelling rhythmic and temporal features at the cost of ignoring the nuances.

[6] This paper will not discuss the velocity feature since it was not involved in the data challenge.

As some officially provided melodies were generated without clear beats due to the flaws of some generation systems, the choice of metric unit can be vital in this challenge.

12th of a quarter note is a commonly used metric unit of time, where durations are multiples of 8th notes and 8th triplets, as default used by the Python package `music21` when opening midi files [5]. 12th was adopted by Li [7] and Xia [9]. 24th was adopted by Yu [14] and Guo [8], as used in the Python package `pypianoroll` [16]. Ding's [10] and Wang's [11] submissions used only 4th, a much wider step, which resulted in much lower AUC scores. However, Deng's system [6] did not use the note duration. These facts and the rankings suggest that the choice of larger metric unit could potentially increase the task difficulty of this challenge due to the loss of rhythmic details.

5.3 Effects of Melody Representations on Model Performance

Another experiment was conducted to support the assumption that melody representation impacts the model performance of this task. The top-ranked system by Li, an LSTM-based binary classifier, was re-implemented so that all the 8 representations aforementioned can be used as the model input. For all the re-implemented and re-trained models, the dataset specification remained unchanged as listed in Table 3. The resulting AUC scores are presented in Table 4 and also at the bottom right corner in the boxes of Figure 3. The quantisation options are selected based on the best system performance. For example, the *Pianoroll-Pitch-Scalar* and the *Pianoroll-Pitch-Embedding* representations must use a 3-time larger quantisation grid compared to others so that the sequence length is decreased from around 400 (8 bars, 4 beats and 12 subdivisions per beat) to 100 (4 subdivisions per beat), which allowed the model to converge. The model failed to converge for representations *Event-PitchDuration-Sequence* and *Tonality*. The former failed probably because the doubled sequence length exceeded the network's capacity. The later, despite being short enough, seemed not to help the model learn advanced tonal features purely from the bar tonality distributions. By Deng *et. al*'s report [6], the bar tonality scores, before being used as features, are adjusted according to the confidence of each bar having a clear tonal center, which may be difficult to be learnt by Li's model.

#	Representation	AUC
1	Event-Pitch-Scalar	0.57
2	**Tonality** [6]	0.48
3	**Pianoroll-Pitch-Scalar** [11]	0.57
4	Event-Pitch-Embedding	0.63
5	**Pianoroll-Pitch-Embedding** [10, 14]	0.70
6	**Event-PitchDuration-Sequence** [8, 9]	0.52
7	**Event-PitchDuration-Addition** [7]	0.87
8	Event-PitchDuration-Concatenation	0.87

Table 4: Different input representations tested on the same system. Bold ones came from the submissions.

The results are basically consistent to the assumption that melody representations will impact the model performance in this data challenge. Notice that *Tonality* and *Event-PitchDuration-Sequential* used on the top-performing model did not result in effective classifiers with the AUC-ROC score being only 0.48 and 0.52, which was possibly due to the limited information represented by the tonality vectors and the over-long sequential representation respectively. Figure 3 combined with the results also to some extent displayed the advantages and drawbacks of different melody representations applied to the same model. This provides a reference to researchers when choosing input representations for their music data.

5.4 Effects of Metric Unit on Model Performance

Another set of experiments were conducted to investigate the influence of metric unit on the model performance. The system is modified to accommodate different unit metrics used with the original representation methods of **Event-PitchDuration-Addition**. The attempted metric units are 8th, 12th, 16th and 24th of a quarter note.

#	Metric Unit	Vocab Size	AUC
1	8th	89(p) + 8(d)	0.83
2	12th	89(p) + 15(d)	0.87
3	16th	89(p) + 16(d)	0.88
4	24th	89(p) + 94(d)	0.86

Table 5: Model performances using different metric units of time. The framed line refers to the original representation used by Li's submitted systems.

The results from Table 5 indicates the influence of metric unit of time on the model performance. When the metric unit is small, more rhythmic nuances are preserved and the corresponding vocabulary size (the number of tokens needed to donate all possible duration [or positional] information in the *training* dataset, in Li's manner [7]) will usually be larger. The accordingly changing AUC-ROC scores suggest the necessity of carefully choosing the metric unit of time for a specific music representation.

6. CONCLUSION

The CSMT 2020 Data Challenge evaluated objective systems that identify generated melodies from human compositions. With no limitation on the methodologies and external dataset, participants designed and submitted a wide range of systems utilising DNNs, heuristic methods, and traditional ML models. According to the challenge results, this paper investigated the influence of music style, pitch features and music representations on the model performance. Minor influence of the two music styles (*Bach* and *Pop*) are observed from the results. However, noticeable impacts are observed regarding pitch features and input representation of melodies. This paper suggests that the pitch features could be effectively used to diagnose the model performance for both music generation systems and objective evaluation systems. Besides, melody representations are also important factors that should be carefully selected in related tasks.

7. ACKNOWLEDGEMENT

The authors acknowledge the contribution from all members in the organisation committee of the CSMT Data Challenge 2020. This work is partially supported by Beijing Zhongwen (Xiamen) Law Firm.

8. REFERENCES

[1] S. Li, Y. Jing, and G. Fazekas, "A Novel Dataset for the Identification of Computer Generated Melodies in the CSMT Challenge," in *Proceedings of the 8th Conference on Sound and Music Technology.* Springer Singapore, Apr. 2021, pp. 177–186.

[2] A. Roberts, J. Engel, C. Raffel, C. Hawthorne, and D. Eck, "A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, Jul. 2018, pp. 4364–4373.

[3] Y.-S. Huang and Y.-H. Yang, "Pop Music Transformer: Beat-based Modeling and Generation of Expressive Pop Piano Compositions," in *Proceedings of the 28th ACM International Conference on Multimedia*. New York, NY, USA: Association for Computing Machinery, Oct. 2020, pp. 1180–1188.

[4] L.-C. Yang, S.-Y. Chou, and Y.-H. Yang, "MidiNet: A Convolutional Generative Adversarial Network for Symbolic-domain Music Generation," *arXiv:1703.10847 [cs]*, Jul. 2017.

[5] M. Cuthbert and C. Ariza, "Music21: A Toolkit for Computer-Aided Musicology and Symbolic Music Data." in *Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR 2010*, Jan. 2010, p. 642.

[6] Y. Deng, Z. Xu, L. Zhou, H. Liu, and A. Huang, "Research on AI Composition Recognition Based on Music Rules," *arXiv:2010.07805 [cs]*, Oct. 2020.

[7] Y. Li and Z. Lin, "Melody Classifier with Stacked-LSTM," *arXiv:2010.08123 [cs, eess]*, Nov. 2020.

[8] J. Guo, A. Liu, and J. Xiao, "Melody Classification based on Performance Event Vector and BRNN," *arXiv:2010.07562 [cs, eess]*, Oct. 2020.

[9] Y. Xia, Y. Jiang, and T. Ye, "Music Classification in MIDI Format based on LSTM Mdel," *arXiv:2010.07739 [cs, eess]*, Oct. 2020.

[10] M. Ding and Y. Ma, "A Transformer Based Pitch Sequence Autoencoder with MIDI Augmentation," *arXiv:2010.07758 [cs, eess]*, Feb. 2021.

[11] Y. Wang, Y. Liu, and L. Zhang, "An AI-Generated Melody Detection System Based on Support Vector Machine," *unpublished*, 2020.

[12] B. L. T. Sturm, M. Iglesias, O. Ben-Tal, M. Miron, and E. Gómez, "Artificial Intelligence and Music: Open Questions of Copyright Law and Engineering Praxis," *Arts*, vol. 8, no. 3, p. 115, Sep. 2019.

[13] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, I. Simon, C. Hawthorne, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, "Music Transformer," *arXiv preprint arXiv:1809.04281*, 2018.

[14] J. Yu, J. Wang, and Y. Zhao, "Identification of AI-Generated Melodies," *unpublished*, 2020.

[15] L.-C. Yang and A. Lerch, "On the Evaluation of Generative Models in Music," *Neural Computing and Applications*, vol. 32, no. 9, pp. 4773–4784, May 2020.

[16] H.-W. Dong, W.-Y. Hsiao, and Y.-H. Yang, "Pypianoroll: Open Source Python Package for Handling Multitrack Pianoroll," in *Pypianoroll: Open Source Python Package for Handling Multitrack Pianoroll.*, 2018.