Araneda-Hernandez *et al.*
EURASIP Journal on Audio, Speech, and Music Processing (2023) 2023:19
https://doi.org/10.1186/s13636-023-00279-6

EURASIP Journal on Audio, Speech, and Music Processing

EMPIRICAL RESEARCH

Open Access

MUSIB: musical score inpainting benchmark

Mauricio Araneda-Hernandez[1,2,3*], Felipe Bravo-Marquez[1,2,3], Denis Parra[2,3,4] and Rodrigo F. Cádiz[3,5]

Abstract

Music inpainting is a sub-task of automated music generation that aims to infill incomplete musical pieces to help musicians in their musical composition process. Many methods have been developed for this task. However, we observe a tendency for each method to be evaluated using different datasets and metrics in the papers where they are presented. This lack of standardization hinders an adequate comparison of these approaches. To tackle these problems, we present MUSIB, a new benchmark for musical score inpainting with standardized conditions for evaluation and reproducibility. MUSIB evaluates four models: Variable Length Piano Infilling (VLI), Music InpaintNet, Music SketchNet, and AnticipationRNN, and over two commonly used datasets: JSB Chorales and IrishFolkSong. We also compile, extend, and propose metrics to adequately quantify note attributes such as pitch and rhythm with *Note Metrics*, but also higher-level musical properties with the introduction of *Divergence Metrics*, which operate by comparing the distance between distributions of musical features. Our evaluation shows that VLI, a model based on Transformer architecture, is the best performer on a larger dataset, while VAE-based models surpass this Transformer-based model on a relatively small dataset. With MUSIB, we aim at inspiring the community towards better reproducibility in music generation research, setting an example for strongly founded comparisons among SOTA methods.

Keywords Music generation, Music inpainting, Music infilling, Benchmark, Evaluation, Reproducibility

1 Introduction

Composing musical pieces is a challenging and complex task. Several computational models have been proposed to help in this human-creative process. Deep learning techniques have emerged as the tool of choice for model design in this field mainly because of their ability to learn complex implicit rules and temporal dependencies in the data [1]. *Musical Score Inpainting* (or *Infilling*) is a sub-task of automated music generation that aims to infill incomplete musical pieces to help musicians in their composition process. In this setting, musicians can easily interact with a model giving incomplete ideas they want to join or parts they want to extend. This setting is powerful since most approaches are based on the sequential music generation paradigm, in which users do not have the flexibility to include their ideas outside the start of the musical piece.

Formally, as defined in [2], the musical inpainting task consists of: *Given a past musical context* C^p, *a future musical context* C^f, *the modeling task is to generate an inpainted sequence* C^m, *which can connect* C^p *and* C^f *in a musically meaningful manner* (Fig. 1).

Several methods have been proposed to address this task. However, we do not observe a consistent and comparable setup among them (i.e., each method is trained and evaluated using different datasets and metrics). Additionally, we identify two main issues with most metrics: (1) they do not measure whether the generated music is adequate for its given context, and (2) their values are dependent on the chosen time discretization. This brings a problem when comparing approaches, hindering

*Correspondence:
Mauricio Araneda-Hernandez
maraneda@dcc.uchile.cl
[1] Department of Computer Science, University of Chile, Santiago, Chile
[2] Millennium Institute for Foundational Research on Data (IMFD), Millennium Institute for Foundational Research on Data (IMFD), Santiago, Chile
[3] National Center for Artificial Intelligence (CENIA), Santiago, Chile
[4] Department of Computer Science, Pontificia Universidad Catolica de Chile, Santiago, Chile
[5] Music Institute and Department of Electrical Engineering, Pontificia Universidad Catolica de Chile, Santiago, Chile

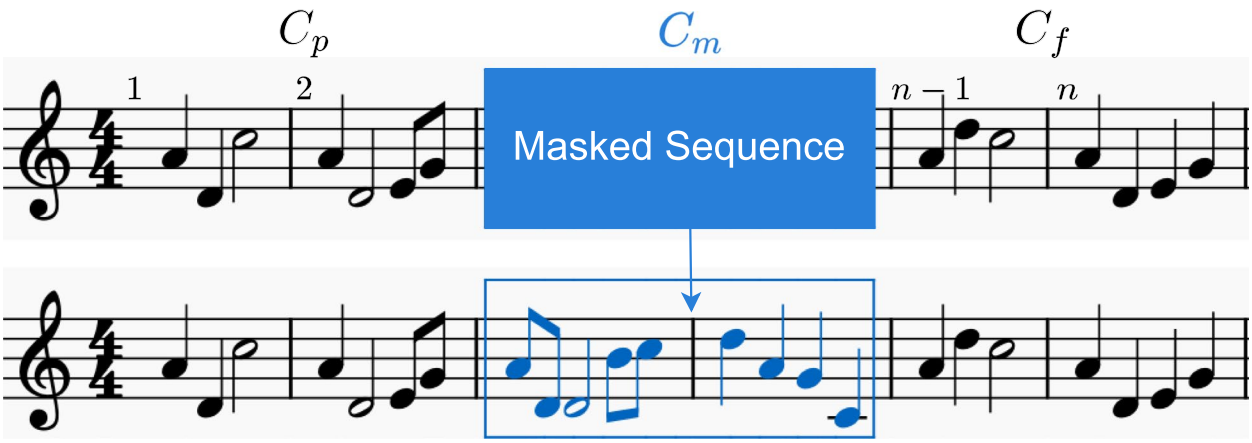


Fig. 1 Music Inpainting Task

the understanding of what is useful or not and what is the state of the art for the task.

With the same spirit, several research communities have highlighted the need for stronger standards on reproducibility to avoid a false sense of progress. For instance, for the task of collaborative filtering recommendation, Dacrema et al. [3] benchmarked several deep learning methods against traditional approaches finding that there was not a clear and consistent improvement of deep learning techniques over matrix factorization methods. Furthermore, Arango et al. [4] found that the task of hate speech detection had made less progress than reported in literature after benchmarking several methods under the same datasets with equal training and testing conditions. These examples support the need for strong, fair and replicable benchmarks to claim progress in a task. These works inspire us to develop MUSIB for music inpainting.

In this paper, we propose MUSIB, a new benchmark initiative for musical score infilling evaluation. Even though several types of non-single monophonic music generation techniques have been developed [5–7], we target single-monophonic music inpainting (i.e., one note per time-step at most) as this task has received significant attention in the literature in recent years and has been shown that its non-sequential approach to music generation is a good approximation of the human compositional process [8].

Our benchmark proposes standardized conditions for reproducibility, ensuring the validation of the correctness of these models while facilitating the identification of the best-suited model for a given purpose/domain.

We compile, extend and propose metrics to measure notes attributes of predicted notes such as pitch, rhythm, and onset position with *Note Metrics*, but also more general attributes such as similarity between infilled sequence and its context with our proposed *Divergence Metrics*.

Our benchmark comprises the evaluation of four models: VLI [9], Music SketchNet [10], Music Inpainting [2] and Anticipation-RNN [11], and over two commonly used datasets: JSB Chorales [12] and IrishFolkSong [13]. Furthermore, our evaluation methodology can be easily extended to new datasets and models.

We list our contributions as follows: (I) To the best of our knowledge, we propose the first Music Inpainting benchmark with standardized datasets and metrics. (II) We propose a new set of task-specific metrics that quantify how adequate is an infilled musical piece to its context. (III) We publicly release a music inpainting library for data processing, training, and predicting with all methods covered in this work[1].

1.1 Related work

In this section, we review related work according to two perspectives: models and metrics. The models discussed are the ones included in our benchmark. There are other methods such as DeepBach [14] and CocoNet [15] focused on multi-monophonic music inpainting that are beyond the scope of this benchmark. In Section 2.4, we discuss some metrics used for evaluation and their current limitations.

1.1.1 Models

We identify three types of architectures that have been used for this task: RNN, VAE, and Transformers, which are discussed below (Table 1):

RNN based Anticipation-RNN [11] model represents input as a sequence of integers encoding each note

[1] https://github.com/maranedah/music_inpainting_benchmark

Table 1 Original evaluation conditions for music inpainting models, showing how difficult is to compare them

Model	Dataset	Metrics	Baseline
VLI [9]	AILabs-1k7	H1, H4, GS	Modified ILM and FELIX
SketchNet [10]	IrishFolkSong	pAcc, rAcc, NLL	InpaintNet
InpaintNet [2]	IrishFolkSong	NLL	A-RNN
A-RNN [11]	JSB Chorales	Accuracy, JS Div	Unconstrained A-RNN

token. Two RNNs capture two temporal sequences: one encodes unary-constraints embeddings while the other auto-regressively generates tokens conditioned on these constraints. Unary constraints allow the model to include pre-defined notes at arbitrary timesteps. Constraining C_p and C_f before the generation process recreates the musical score inpainting setup.

VAE based Music InpaintNet [2] use VAEs [16] to encode isolated monophonic measures into a latent space vector (MeasureVAE). The model uses this normalized space to operate the past z_p and future z_f with a LatentRNN. This last representation is then hierarchically decoded into beats and ticks for reconstructing the inpainted sequence. Another approach, Music SketchNet [10], uses a similar strategy to [2] using VAEs to encode measures. However, it modifies the data representation to naturally separate pitch and rhythm into two separate dimensions. Their VAE encodes these two separate channels and then decodes them hierarchically. A final training phase is done to condition the final output with users' input over general constraints on pitch and rhythms.

Transformer based Variable Length Infilling (VLI) [9] proposed a model for single-polyphonic music inpainting based on XLNet [17]. This method encodes each note event as a word token and feeds it to a pre-trained language model. They incorporate a musically specialized positional encoding called relative bar encoding to keep track of the relative position of each note within its context.

1.1.2 Metrics

Designing good metrics for music generation is still an open problem. The most limiting factor is that given a seed or a constraint, music often has a variety of solutions that would fit in a given scenario.

Negative Log-Likelihood (NLL) has been widely used both for training and evaluating models on music inpainting. This value represents a statistical distance between two given distributions, where the lower the value, the closer these distributions are. The main drawback of this function is that the value itself does not represent any musical concept nor captures the domain's semantics and thus cannot be analyzed intuitively.

Two simple yet intuitive metrics were proposed by Chen et al. [10] called *pAcc* and *rAcc*. They represent whether the model generates the correct pitch/rhythm token in the correct time-step position. However, pAcc has limitations in distinguishing, for example, whether a note is misplaced or is different from the expected pitch, while rAcc can change its values depending on chosen time step discretization.

Pitch Class Histogram Entropy and *Grooving Pattern Similarity* [18] were used in [9] to compare the similarity of musical attributes between measures in the infilled part with those present in the context. These metrics were used assuming that to generate fluent music, the metrics calculated on C_m should be close to the ones calculated for C_p and C_f. They state that the lower the differences between the middle part metrics and its context, the better the model is. However, this is not necessarily true since values too close to zero indicate that the model is just repeating its context instead of articulating past and future musical ideas.

Finally, some other metrics capture more general musical attributes that are not directly comparable between two sequences but can be useful in understanding the output of a model. For example, *Number of Silence* [19] calculates the percentage of empty time steps in a sequence to check if a model is generating too much silence or not.

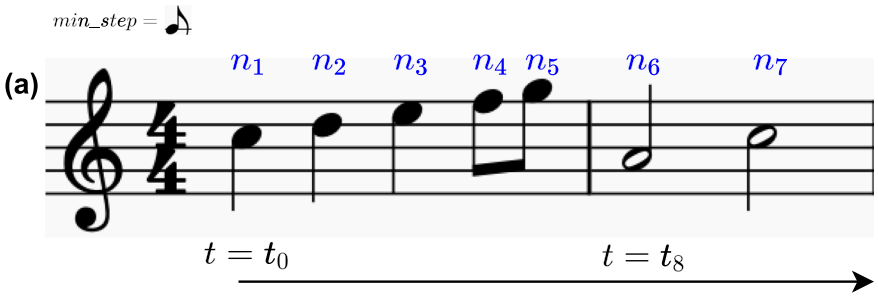
2 Methods

In this section, we present the methods for MUSIB, our proposed musical score inpainting benchmark studied in this empirical research, which comprise models, data sets and evaluation metrics.

2.1 Models

The models considered for evaluation are VLI [9], SketchNet [10], InpaintNet [2], and Anticipation-RNN [11]. Each model was trained from scratch, setting its hyperparameters as defined on its source implementation. This includes the number of layers, hidden size, optimizer, learning rate, and dropout, among others.

The data processing pipeline of each model is reproduced from its corresponding source code implementation. The data representation that is fed into each model is shown in Fig. 2.



(b) $x = [C_4, \_, D_4, \_, E_4, \_, F_4, G_4, A_3, \_, \_, \_, C_4, \_, \_, \_]$

(c)
$$x = (x_{pitch}, x_{rhythm})$$
$$x_{pitch} = [C_4, D_4, E_4, F_4, G_4, A_3, C_4]$$
$$x_{rhythm} = [1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0]$$

(d) $x = [\ [n_1, n_2, n_3, n_4, n_5], [n_6, n_7]\ ]$

n	Tempo	Bar Start	Position	Pitch	Velocity	Duration
n_1	120	1	0 | 8	C_4	90	
n_2	120	0	2 | 8	D_4	90	
n_3	120	0	4 | 8	E_4	90	
n_4	120	0	6 | 8	F_4	90	
n_5	120	0	7 | 8	G_4	90	
n_6	120	1	0 | 8	A_3	90	
n_7	120	0	4 | 8	C_4	90	

Fig. 2 Data representation for each method. For easier visualization, temporal resolution is set as two time-steps per quarter note. **a** Musical input data. **b** Vectorial representation used in Anticipation-RNN [11] and Music InpaintNet [2]. Data is represented as a temporal sequence array of *t* time steps. The token "_" represents the state of holding a note. **c** Representation used in Music SketchNet [10]. The input is separated ("factorized") into pitch and rhythm dimensions. **d** Representation used in VLI [9]. It captures each note event and groups them into a list of measures

2.2 Datasets

We selected JSB Chorales and IrishFolkSong datasets to implement our evaluation. We prioritized these datasets since they have been used to train several musical inpainting models. They also represent different musical styles which provide meaningful differences in their musical content. Finally, they have an important difference in size, making generalizing results a challenging task.

JSB Chorales dataset contains 408 samples of 4-voices chorales pieces. Each sample corresponds to an harmonization of a hymn. Its source format is MXL; however, we transform the data to MIDI format to have a single pipeline to process all data.

IrishFolkSong [13] dataset contains 45,849 pieces of monophonic folk tunes in midi format.

To gain insight into the differences in the melodic properties of the two datasets, we have added their corresponding pitch histograms to our Section 2.2. They reveal significant differences in their tonality patterns. In particular, we observe that JDB exhibits higher variance in tonalities than IrishFolk, which we attribute to a higher complexity in melodic patterns.

An overview of the pitch distribution for both dataset is shown in Figs. 3 and 4. It can be observed from this that both datasets reveal significant differences in their tonality patterns. In particular, we observe that JSB Chorales exhibits higher variance in tonalities than IrishFolk,

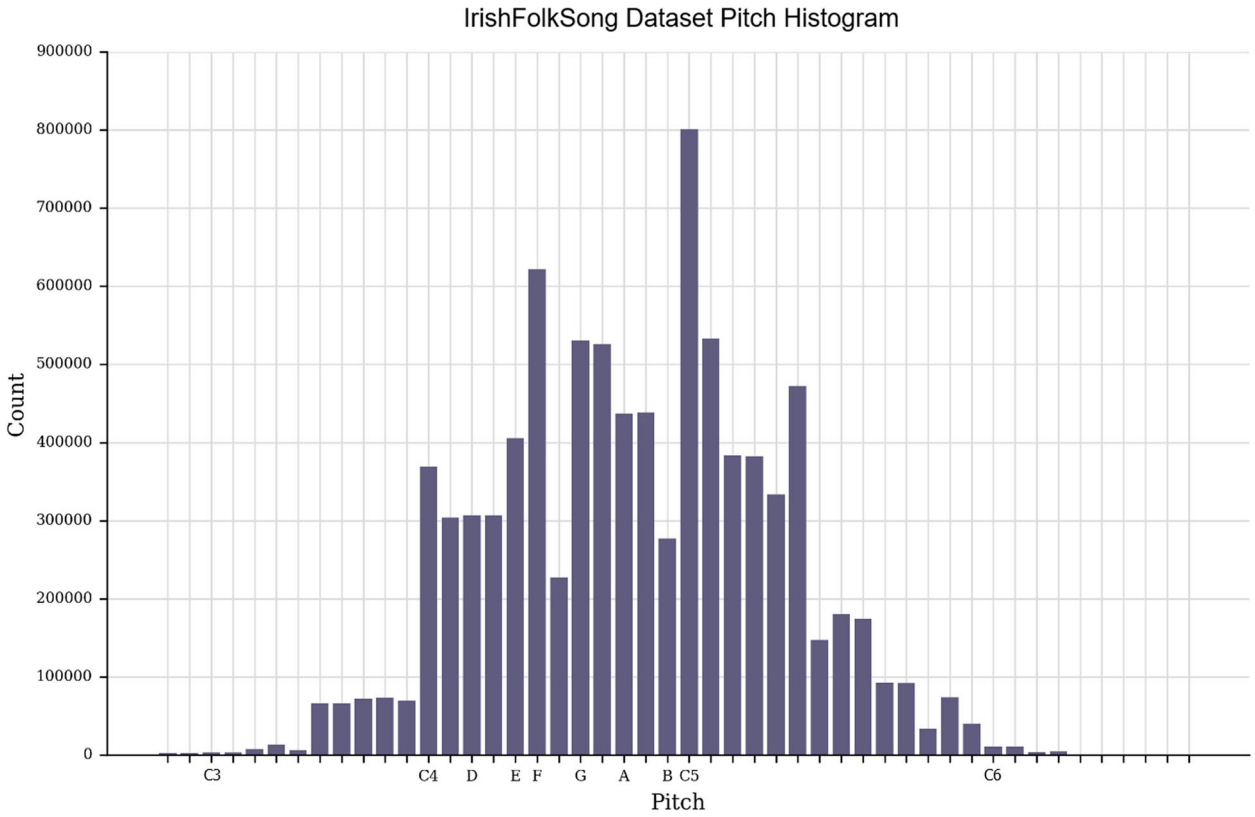


Fig. 3 Pitch distribution for the IrishFolkSong dataset

which we attribute to a higher complexity in melodic patterns.

For both datasets, we filtered invalid files (i.e., no instruments or zero-length), repeated files (files with the same hash), and files shorter than 16-measures long. We also filtered the data in IrishFolkSong to only have pieces on 4/4 time signatures. This last step is done to reproduce conditions described in papers that originally used this dataset.

The final JSB Chorales dataset contains 171 songs, totaling 13,304 measures that were grouped into 2360 contexts. IrishFolkSong dataset ended up with 17,538 songs, 605,164 measures, and 324,556 grouped contexts.

2.3 Experimental Setup

We fixed each context size to be 16-measures long, C_p and C_f are 6-measures long, while C_m is 4-measures long. Each measure is 4-bar long. Each measure is discretized by 24 time steps for all the models, making each of this context to be 384 time-steps long.

For each dataset we split the full set of songs into train, validation, and test sets with an 8:1:1 ratio. These sets were fixed during the evaluation and will be publicly available for reproducibility. During the training phase, for each epoch, we randomly cropped each song to be 16-measures long to group a context. We used Early Stopping with a patience of 5 epochs.

2.4 Metrics

In MUSIB, we compile the following metrics: *Negative Log-Likelihood* (NLL), *Pitch Accuracy* [10], *Rhythm Accuracy* [10], *Pitch Class Histogram Entropy* [18], *Groove Similarity* [18], and *Number of Silences* [19]. Additionally we propose *Position Score* to introduce a new evaluation dimension for the onset of notes.

We modified *Pitch Accuracy* and *Rhythm Accuracy* to receive different musical representations while preserving consistency in results.

We additionally extended *Pitch Class Histogram Entropy*, *Groove Similarity*, and *Number of Silences* to be calculated as *Divergence Metrics*.

We classify MUSIB metrics into two groups: *Note Metrics* and *Divergence Metrics*.

2.4.1 Note metrics

Note metrics directly compare notes attributes in predicted data vs true data, one note at a time. We argue that for measuring the quality of notes predicted, we need to

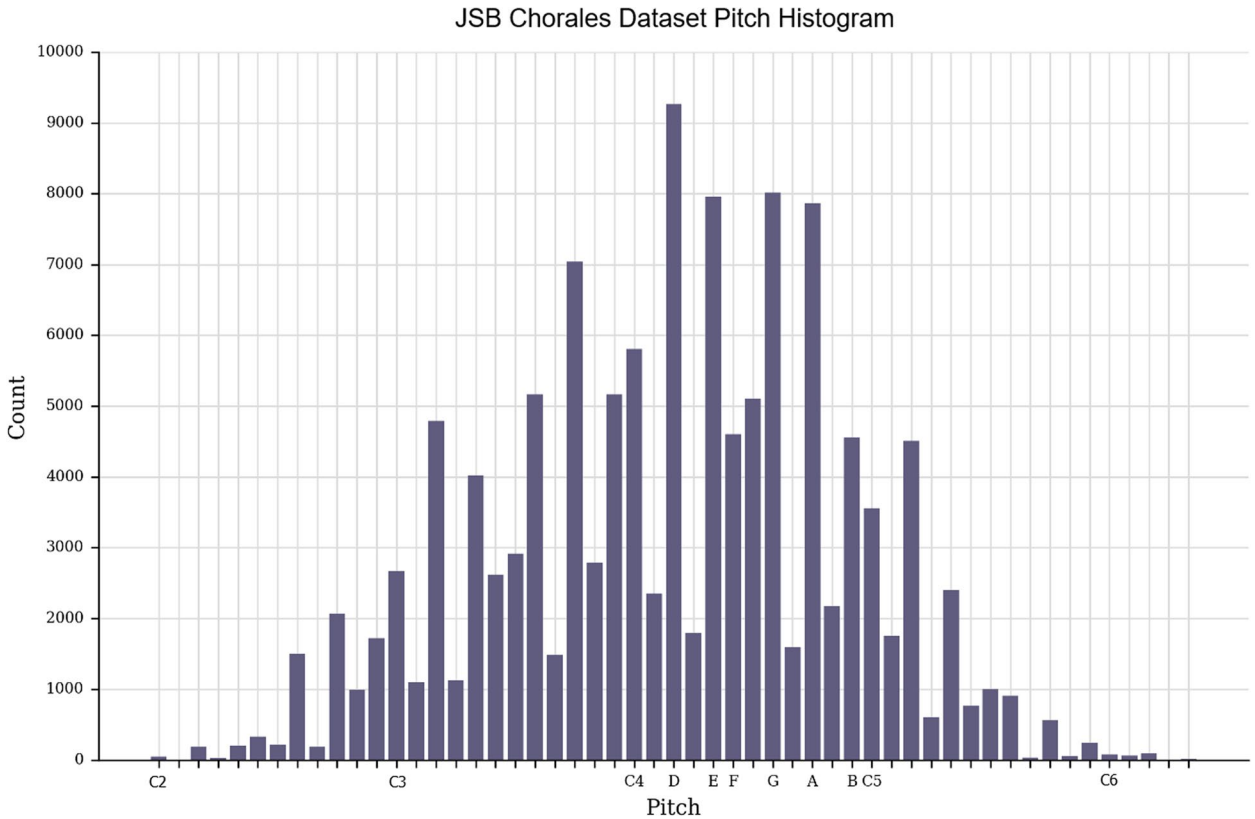


Fig. 4 Pitch distribution for the JSB Chorales dataset

compare at least three dimensions: Position, Pitch, and Rhythm. Position indicates the time where a note starts, Pitch is the pitch of a note (C_4, G_2, $D_3^{\#}$, ...), and Rhythm is the combination of duration and position of a note. We represent all notes in Y_{true} and Y_{pred} as triplets for these three dimensions when evaluating note metrics.

Position Score Position Score is a metric proposed in this work that measures the similarity of two musical sequences in terms of the position of their notes.

We argue that to correctly measure notes' position similarity, a metric needs to be able to meet the following requirements:

1. Be equipped with a strategy to align the notes' positions within gold and predicted sequences independently of the order in which they appear.
2. Handle sequences with potentially different number of notes.
3. Reward sequences that share the same positions for their notes.
4. Penalize sequences that do not share the same positions for their notes.
5. Penalize generated sequences with different number of notes than expected.

Delving deeper into requirement (1), we should point out that the *i*th note of the gold sequence may be present as the *j*th note of the predicted sequence. Therefore, to check that a given position has been correctly predicted, it is important that our metric can align the positions between the two sequences to perform a proper evaluation.

Taking the above into account, we construct our metric as an F1 score calculated from gold and predicted note's positions whose internal variables (i.e., True Positives, False Positives, False Negatives) are computed as follows:

- True Positives (TP): A note's position is present in both sequences.
- False Positives (FP): A note's position is present in the generated sequence when it was not present in the gold sequence.
- False Negatives (FN): A note's position is missing in the generated sequence when it was present in the gold sequence.

Note that True Negatives are not part of the F1 score function and thus its definition is not stated here. Next, we discuss how each of the the aforementioned requirements are satisfied by our F1 metric:

1. By defining the process of alignment based on checking the presence of a note within a given sequence, we resolve the ordering problem between non-matching sequences.
2. Building the internal variables of the F1 Score based on the alignment of positions allows us to compare sequences with different number of notes since the match of positions for the *i*th and *j*th note may occur at arbitrary indexes in arbitrary long sequences.
3. Both values *precision* and *recall* will increase as the number of True Positives increases, increasing F1-Score performance, and thus rewarding sequences that share positions.
4. Both values *precision* and *recall* will decay as *FP* and *FN* increase. Note that metric functions such as *Accuracy* would not be able to penalize missing notes (*FN*). Additionally, there is no difference in cost for different types of mis-classifications in this task. Either adding or removing notes to the generated sequence with respect to the gold sequence would have the same impact in musicality. Due to this, both the recall and precision do not need particular weights when being evaluated, discarding alternatives such as F_β functions.
5. If the generated sequence contains more notes than the true sequence, the number of false positives will increase. Similarly, if the number of notes is smaller than the true sequence, the number of false negatives will increase. Both cases imply that F1-Score will decrease in performance, either by a worse Recall or Precision. This implies that *Position Score* penalizes sequences with a different number of notes than expected.

We formally define *Position Score* as:

$$pos_{F1}(y,\hat{y}) = F_{1_{(tp,fp,fn)}}(y,\hat{y})$$

$$tp(y,\hat{y}) = \sum_{i=0}^{n} \mathbb{1}_{y_i \in \hat{y}}$$

$$fp(y,\hat{y}) = \sum_{j=0}^{m} \mathbb{1}_{\hat{y}_i \notin y}$$

$$fn(y,\hat{y}) = \sum_{i=0}^{n} \mathbb{1}_{y_j \notin \hat{y}}$$

where y is the list of positions in the gold sequence, $\hat{y}$ is the list of positions in the predicted sequence, *tp* is the function that computes true positives, *fp* is the function that computes false positives, *fn* is the function that computes false negatives, n is the number of notes present in the gold sequence, m is the number of notes present in the predicted sequence, and $F_{1_{(tp,fp,fn)}}(y,\hat{y})$ is the f1 score computed from the result of the *fp*, *tp*, and *fn* functions applied over y and $\hat{y}$.

Pitch Accuracy Firstly defined by Chen et al. [10], is the percent of pitches correctly predicted over the total of pitches in a sequence. The metric is thought as a comparison of two musical sequences, where if a pitch is present at a given time index, the metric function checks the equality of this pitch in the same index for the other sequence.

We argue that this metric, in its current form, may be misleading in explaining two fundamentally different musical phenomena since a mismatch of pitch might represent either:

1. The first note and the note to be compared (both at time index i) do not share the same pitch (e.g., one note is F3 and the other one is D4), or
2. There is a note at time index i for the first sequence, but there is no matching note at the same time index in the sequence to compare because there is a silence or hold token.

The second case is a case of misplacing of notes instead of an error of pitches. In Fig. 5a, we show an example where the two different phenomenon lead to the same result.

To address this ambiguity, we restrict the instances to which this metric is applied to only pairs of notes with matching positions within their corresponding sequences; otherwise, the comparison is omitted. The intuition is that notes that share position but not pitch will be measured by *Pitch Accuracy*, while the misplaced notes will be measured by *Positional Score*. We argue that this modification gives a clearer understanding of the output of the *Pitch Accuracy* metric, addressing potential concerns when comparing against *Positional Score*. We show an example in Fig. 5b.

Formally, *Pitch Accuracy* is defined as:

$$pAcc(y,\hat{y}) = \frac{1}{N} \sum_{i=0}^{N} \sum_{j=0}^{N} \mathbb{1}_{pitch(y_i)=pitch(\hat{y}_j)} \mathbb{1}_{pos(y_i)=pos(\hat{y}_j)}$$

where N is the number of notes that share positions in both sequences, y is the gold sequence of notes, $\hat{y}$ is the

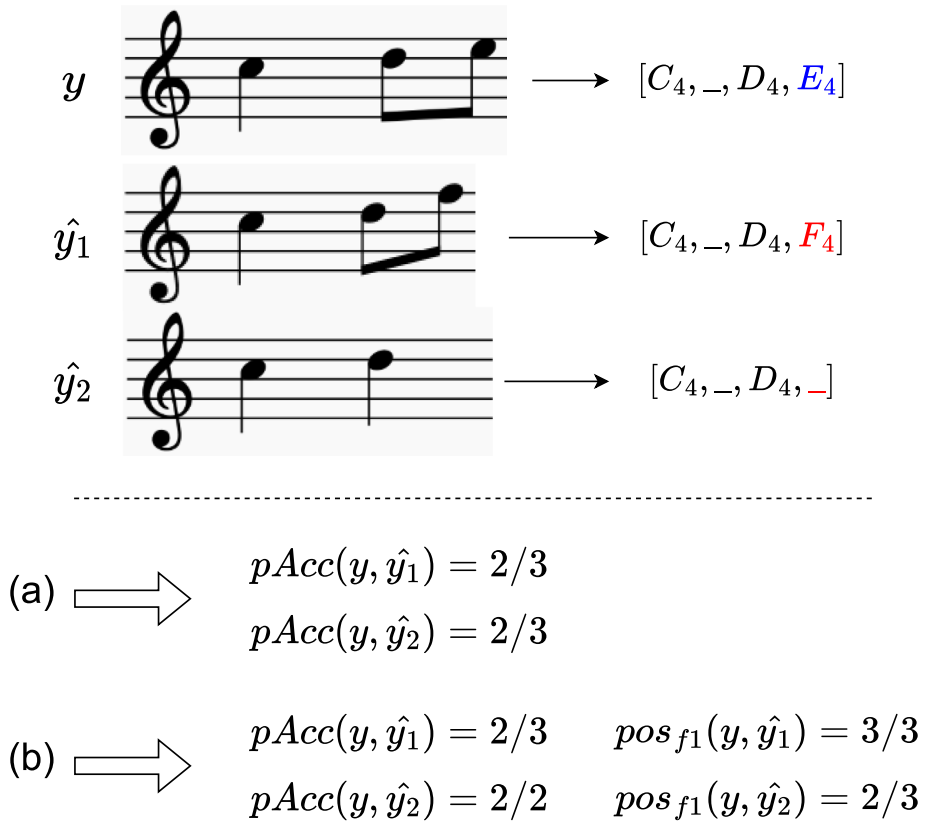


Fig. 5 Example of evaluation between an expected sequence y and two generated sequences $\hat{y}_1$ and $\hat{y}_2$. **a** The results when applying *Pitch Accuracy* over time indexes as proposed by Chen et al. [10]. **b** The results when applying our proposed modification to *Pitch Accuracy* in conjunction with our proposed *Position Score*

predicted sequence of notes, $pitch(y_i)$ retrieves the pitch value of the note at index i, $pos(y_i)$ retrieves the position of the note at index i. Note that the metric definition sums the number of notes that match both pitch and position and then normalizes it by the number of notes present in both sequences.

where N is the number of notes that share positions in both sequences y is the true data, $\hat{y}$ is the predicted data, and y_i^p is the pitch list of the i-th sequence in y.

Rhythm Accuracy Firstly defined by Chen et al. [10], is the percent of notes' duration correctly predicted over the total of notes.

We argue that this metric as it is does not correctly measure the performance of the models due to differences in the results when it is applied to the same data with different notes' resolutions. We show an example in Fig. 6.

Note that the issue comes from the fact that the duration of a note is implied over multiple tokens (one per time-step). Changing the resolution of the sequence affects the representation of hold/silence classes while keeping intact the number of pitch classes. This unbalances the overall distribution and raises errors where rhythm tokens are confused with pitch tokens.

In order to fix this behavior we need to transform the input data before applying the metric such that the rhythm is a single value attached to a note instead of multiple values distributed among multiple time steps. This can be done by representing each note as Note-based discretization including the number of time-steps that a note is held as the rhythm value. The comparison then is applied similarly to *Pitch Accuracy*, where if two notes match in position, then the rhythm values of both notes are compared, otherwise the comparison is skipped and falls under *Position Score* evaluation.

Formally, *Rhythm Accuracy* is defined as:

$$rAcc(y, \hat{y}) = \frac{1}{N} \sum_{i=0}^{N} \sum_{j=0}^{N} \mathbb{1}_{duration(y_i)=duration(\hat{y}_j)} \mathbb{1}_{pos(y_i)=pos(\hat{y}_j)}$$

where N is the number of notes that share positions in both sequences, y is the gold sequence of notes, $\hat{y}$ is the

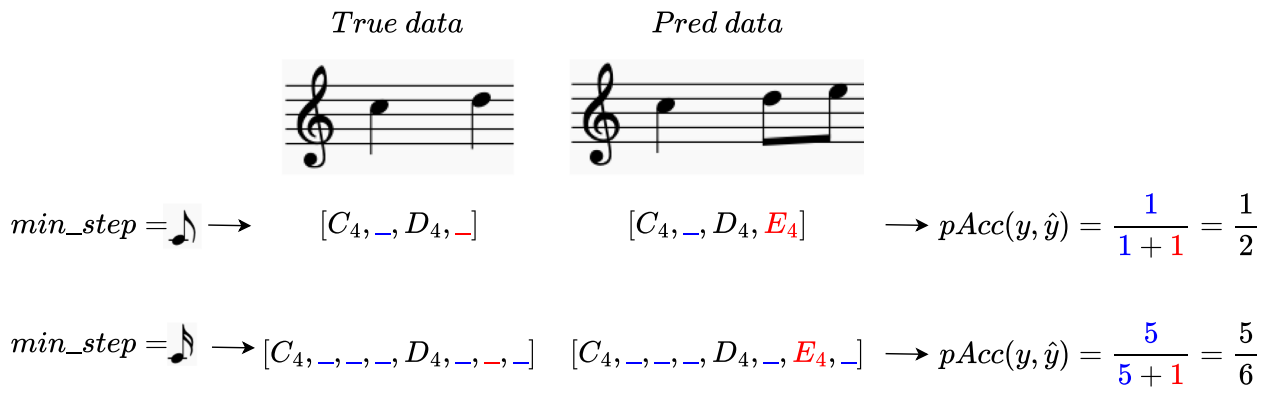


Fig. 6 Example of Rhythm Accuracy giving different results when applied to the same data with different resolution

predicted sequence of notes, $duration(y_i)$ retrieves the pitch value of the note at index i, $pos(y_i)$ retrieves the position of the note at index i. Note that the metric definition sums the number of notes that match both duration and position and then normalizes it by the number of notes present in both sequences.

An example of these three metrics, including the data representation proposed for their application is shown in Fig. 7.

2.4.2 Divergence metrics

Although note metrics are useful for one-on-one comparison, there are cases in music generation where the attributes can not be directly compared since there are multiple correct options.

This variability in music is common and even desirable. However, there is a lack of methods to measure the correct variability of these attributes in generated data.

How do we verify that a given musical attribute in a set of predicted songs is within the correct range of variability? We argue that we need to look at the distribution of this attribute in true data and measure how close it is to the one in generated data. By measuring this closeness between distributions we relax the condition of correctness to accept multiple valid answers.

We introduce *Divergence Metrics* to implement this measuring procedure by defining the divergence of a metric function $f : (x_0, x_1, \cdots, x_n) \rightarrow [0, 1]$ between true data Y_{true} and predicted data Y_{pred} as:

$$f_{div}(Y_{true}||Y_{pred}) = JS_{div}(\overrightarrow{h}_{f(Y_{true})}||\overrightarrow{h}_{f(Y_{pred})})$$

where $\overrightarrow{h}_{f(A)}$ is the histogram of the function values when applied to all sequences in a set and JS_{div} is the Jensen-Shannon Divergence [20, 21]. In our evaluation, we set $\overrightarrow{h}_{f(A)}$ to have 100 bins. This concept is illustrated in Fig. 8.

Note that the metric function f maps a whole sequence to a single value. Since the value of the metric f changes from piece to piece, $f(A)$ will compute a distribution of values given a set A. Therefore, to compare the distribution of a metric in generated data against the one in true data we use a divergence. We choose JS_{div} since it is bounded, symmetric and do not require matching supports [22].

Silence Density Divergence quantifies if the predicted data contains the right amount of silence when compared to the distribution of silence in true data. It is formally defined as:

$$S_{div}(Y_{true}||Y_{pred}) = JS_{div}(\overrightarrow{h}_{S(Y_{true})}||\overrightarrow{h}_{S(Y_{pred})})$$

$$S(x) = \frac{1}{T}\sum_{t=0}^{T} \mathbb{1}_{n\_notes(x_t)=0}$$

where T is the total time steps in the sequence, and $n\_notes(\cdot)$ is the function that counts the number of notes played at a given time step.

Pitch Class Divergence quantifies how similar is the pitch entropy in C_m and $\{C_p \cup C_f\}$. This comparison is done by computing the *Pitch Class Histogram Entropy* according to the notes pitch classes (i.e., C, C#, ..., A#, B) as defined in [18] for each isolated measure. Then we calculate the mean difference between pitch entropy in C_m and pitch entropy in $\{C_p \cup C_f\}$. We formally define H_{div} as:

$$H_{div}(Y_{true}||Y_{pred}) = JS_{div}(\overrightarrow{h}_{H(Y_{true})}||\overrightarrow{h}_{H(Y_{pred})})$$

$$H(x) = \frac{1}{n_1 n_2}\sum_{i=0}^{n_1}\sum_{j=0}^{n_2} |\mathcal{H}_{m_i} - \mathcal{H}_{m_j}|$$

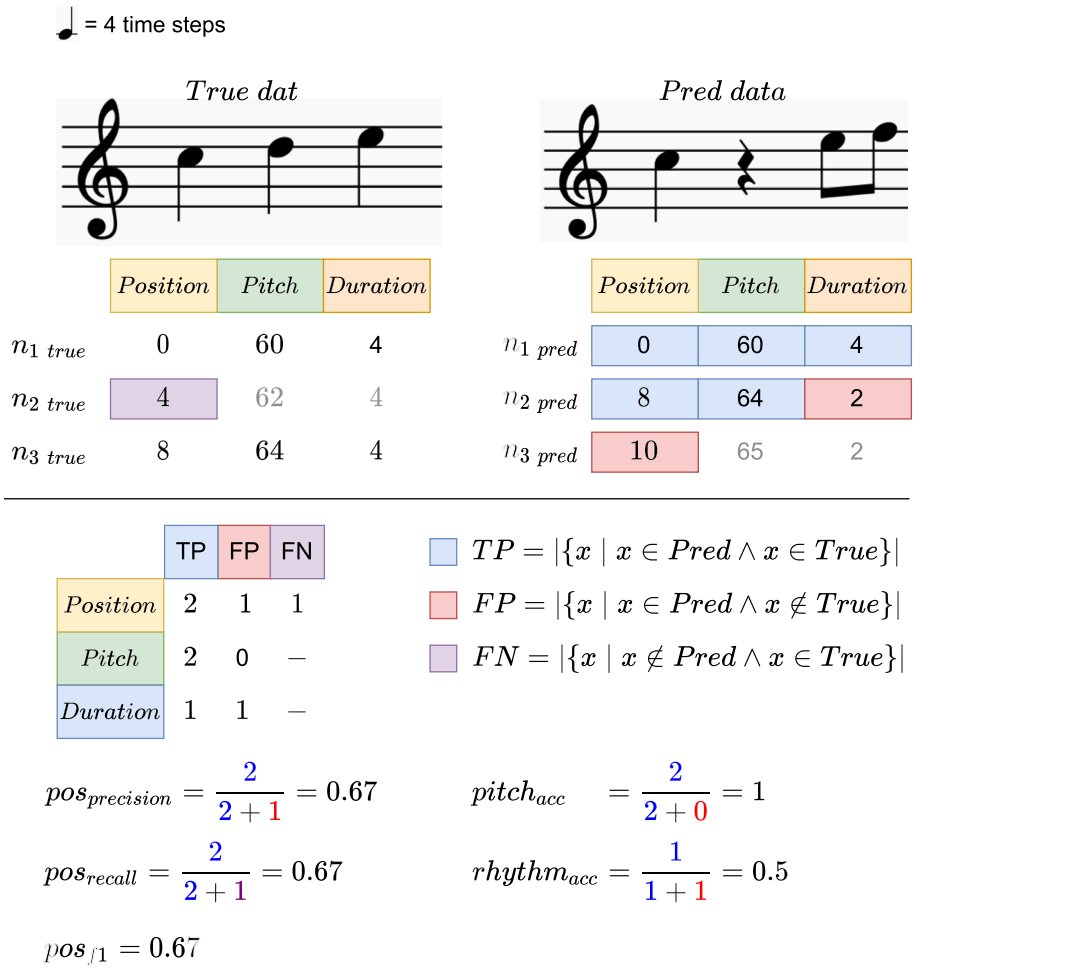


Fig. 7 Note metrics evaluation pipeline. We represent each note in true and predicted data as triplets (Position, Pitch, Duration). We compute true positives, false positives, and false negatives for predicted positions. Then we calculate the position-F1 score, pitch accuracy, and rhythm accuracy. Since we can only compare notes present on both sets, we filter false positives and false negatives when calculating pitch and rhythm accuracy

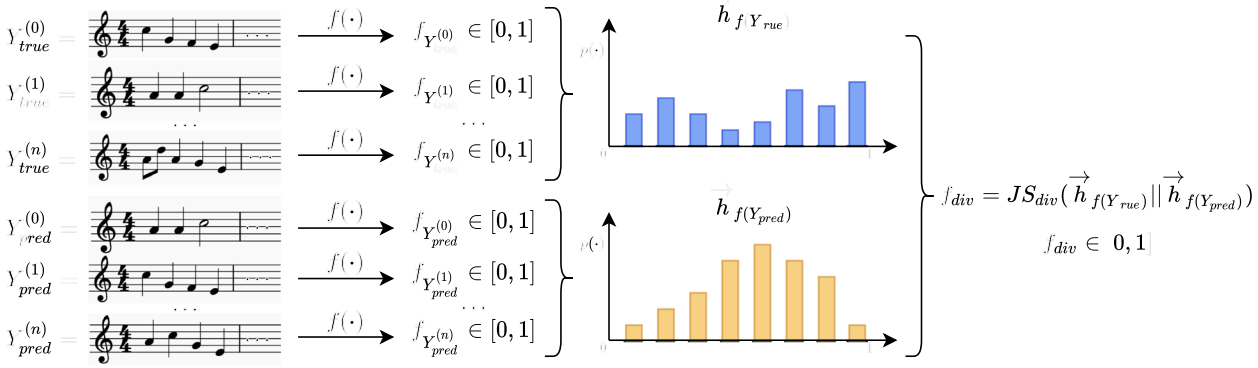


Fig. 8 Divergence Metric of an arbitrary function *f*. Each sequence in Y_{true} and Y_{pred} is mapped to a single value in [0, 1]. Then, the distribution of these values for each set is compared using Jensen-Shannon Divergence

Table 2 MUSIB evaluation on IrishFolk Dataset

IrishFolk Dataset (≈300K samples)							
Model	*NLL* ↓	pos_{F1} ↑	*pAcc* ↑	*rAcc* ↑	S_{div} ↓	H_{div} ↓	GS_{div} ↓
Anticipation-RNN	0.453 (*0.662)	0.930	0.657	0.860	0.017	0.060	**0.007**
InpaintNet	0.487 (*0.662)	0.860	0.517	0.750	0.013	0.174	0.024
SketchNet	0.539 (*0.516)	0.914	0.560	0.868	**0.005**	0.134	0.009
VLI	0.061	**0.940**	**0.861**	**0.952**	0.022	**0.020**	**0.007**

Values in boldface highlight the best performance among the four models on each metric

$$\mathcal{H}_{m_i} = \mathcal{H}(\overrightarrow{h}_{pitch(m_i)}) = -\sum_{i=0}^{11} h_i \log_2(h_i)$$

where n_1 is the number of measures in C_m, n_2 is the number of measures in $\{C_p \cup C_f\}$, m_i are the measures in C_m, m_j are the measures in $\{C_p \cup C_f\}$, and $\overrightarrow{h}_{pitch(\cdot)}$ is the pitch class histogram of a measure.

Groove Similarity Divergence measures how similar are the groove patterns between C_m and $\{C_p \cup C_f\}$. The comparison is made by representing each measure as a sequence of time steps, where 1 corresponds to the start of a note and 0 a hold or a rest. Then the two sequences are compared by penalizing each unmatched value with an XOR function. Similar to *Pitch Class Divergence*, we calculate the mean difference between groove similarities in C_m and $\{C_p \cup C_f\}$.

$$GS_{div}(Y_{true}||Y_{pred}) = JS_{div}(\overrightarrow{h}_{GS(Y_{true})}||\overrightarrow{h}_{GS(Y_{pred})})$$

$$GS(x) = \frac{1}{n_1 n_2}\sum_{i=0}^{n_1}\sum_{j=0}^{n_2}\mathcal{GS}(\overrightarrow{g}^{m_i}, \overrightarrow{g}^{m_j})$$

$$\mathcal{GS}(\overrightarrow{g}^{a}, \overrightarrow{g}^{b}) = 1 - \frac{1}{T}\sum_{t=0}^{T-1} XOR(g_t^a, g_t^b)$$

where n_1 is the number of measures in C_m, n_2 is the number of measures in $\{C_p \cup C_f\}$, m_i are measures in C_m, and m_j are measures in $\{C_p \cup C_f\}$, and $\overrightarrow{g}^{m_i}$ is the encoding of a measure as a rhythm sequence where 1 is an onset, while 0 represents hold or silence.

3 Results and discussion

In Tables 2 and 3, we present the results of all four models evaluated on the IrishFolk and JSB Chorales Dataset, respectively. Arrows indicate if higher/lower values represent better performance. Values in parenthesis are evaluation results declared in the literature. Note that VLI's NLL value is not comparable to the rest since the class prediction setup is encoded differently.

From Table 2, we can observe in the NLL metric that the results declared in the literature are not exactly reproduced in our experiments. In particular, the result for Anticipation-RNN, InpaintNet, and SketchNet, shows a percentual difference of 32%, 26%, and 4%, respectively. We explain this behavior by two variables: hyperparameters and split sets. The reproduction of the models was performed by utilizing the hyperparameters defined on the publication of each model although the hyperparameters observed on the official projects' source code were different, causing potential inconsistencies. Additionally, since the split sets were not publicly available we defined our own sets for training, validation, and testing.

We observe that VLI is the best performer for IrishFolk Dataset in all metrics except for S_{div} while InpaintNet is the best performer on JSB Chorales Dataset across all metrics.

As seen in Figs. 5 and 6, all models have a significant drop in performance from one dataset to another, being the only exception the InpaintNet model for the rAcc metric. The model with the biggest drop in performance is VLI. This result is expected as it has been documented in the literature that transformers models require larger

Table 3 MUSIB evaluation on JSB Chorales Dataset

JSB Chorales Dataset (≈2.4K samples)							
Model	*NLL* ↓	pos_{F1} ↑	*pAcc* ↑	*rAcc* ↑	S_{div} ↓	H_{div} ↓	GS_{div} ↓
Anticipation-RNN	0.459	0.832	0.243	0.682	0.240	0.525	0.232
InpaintNet	0.327	**0.852**	**0.505**	**0.788**	**0.059**	**0.411**	**0.153**
SketchNet	0.605	0.833	0.272	0.708	0.079	0.529	0.228
VLI	0.935	0.643	0.163	0.639	0.197	0.585	0.225

Values in boldface highlight the best performance among the four models on each metric

Table 4 MUSIB evaluation on Small-IrishFolk

Small IrishFolk Dataset (≈2.4K samples)							
Model	*NLL* ↓	pos_{F1} ↑	*pAcc* ↑	*rAcc* ↑	S_{div} ↓	H_{div} ↓	GS_{div} ↓
Anticipation-RNN	0.808	0.879	0.295	0.744	0.071	0.485	**0.163**
InpaintNet	0.655	0.864	**0.419**	0.690	**0.055**	0.489	0.392
SketchNet	0.779	**0.937**	0.386	**0.888**	0.562	0.527	0.176
VLI	0.415	0.851	0.371	0.880	0.102	**0.474**	0.307

Values in boldface highlight the best performance among the four models on each metric

datasets to generalize properly [23]. However, in order to properly decouple the effect of the dataset size when comparing the performance of the models across datasets, we conducted an additional experiment in which we reduced the size of IrishFolkSong to be equal to that of JSB Chorales (171 songs), which we called Small-IrishFolkSong. Results are shown in Table 4.

This new experiment brings the results between JSB Chorales and Small-IrishFolkSong considerably closer, supporting the claim that the size of the datasets greatly affects the performance of the models, even when they have different musical properties. That said, the performance of Small-IrishFolkSong tends to be better than JSB Chorales despite having the same size. We attribute this fact to the complexity of the melodic patterns discussed above. In summary, these results allow us to conclude that both the size of the training set and the complexity of the music affect the performance of the models.

Although InpaintNet achieves the best performance in JSB Chorales, the results roughly surpass 50% pitch accuracy. This exhibits a significant gap in performance for this dataset in contrast to IrishFolkSong that is yet to be solved. Interestingly, from Figs. 5 and 6, we note that InpaintNet has the most stable performance across datasets, having the lowest variation in results of all methods when testing them over IrishFolkSong and JSBChorales.This property may be helpful to improve the stability of other models when there is less data available, and thus improving the task in general.

SketchNet is good at reproducing silence distributions, as seen in S_{div} metric on both datasets, surpassing VLI in IrishFolkSong and seconding InpaintNet in JSB Chorales. This may be explained by SketchNet design. Its representation of data explicitly separates rhythm from pitch, which may help the model focus more on rhythmic patterns and, consequently, on tokens of silence.

VLI is the best model for resembling the distribution of pitch classes between infilled data and its context, as seen in H_{div} metric. This may be explained by its XLNet-based architecture, which would make the model learn the correct pitch distribution first before the correct sequential order of the notes.

Looking at pos_{f1}, *pAcc*, and *rAcc* values across all models we can see that models consistently perform the best in pos_{f1} metric compared to other note metrics. Similarly, all models perform better in *rAcc* than in *pAcc* independent of the dataset. We theorize that music inpainting models first learn to predict the correct onsets of notes, then learn the rhythm, and finally the pitch. However, further experimentation needs to be done in order to verify this hypothesis (Figs. 9 and 10).

4 Conclusions and future work

In this paper, we have proposed MUSIB, a new benchmark for musical score inpainting evaluation. We publicly released our experiments, models, data and code for easier reproducibility[2]. We compiled, extended and proposed metrics to measure meaningful musical attributes in generated data. In particular, we defined two approaches for measuring musical score inpainting performances: Note Metrics and Divergence Metrics. The first relies on one-on-one comparison of note attributes while the second relies on the comparison of distributions between the original dataset and the artificially generated data.

The results obtained from our benchmark suggest interesting findings regarding the state-of-the-art of the musical inpainting task:

1. It was not possible to exactly reproduce the results declared on the literature, having differences ranging from 4 to 32%. This suggests that there is still room for improvement in making the musical inpainting models more reproducible.
2. The performance of existing models is highly dependent on the amount of training data: while VLI, a transformer-based model, achieves the best results when more data is available, and InpaintNet, a VAE-based model, excels when less data is available.

[2] https://github.com/maranedah/music_inpainting_benchmark

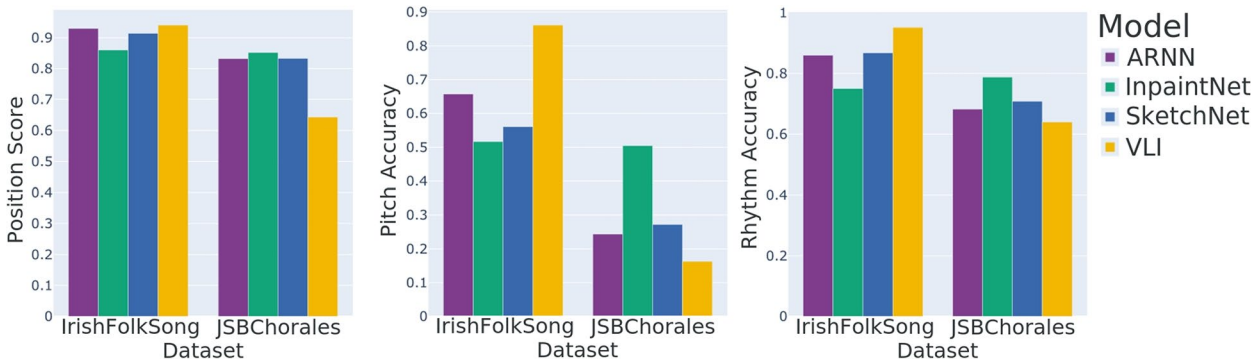


Fig. 9 Comparison of note metrics for IrishFolkSong and JSBChorales

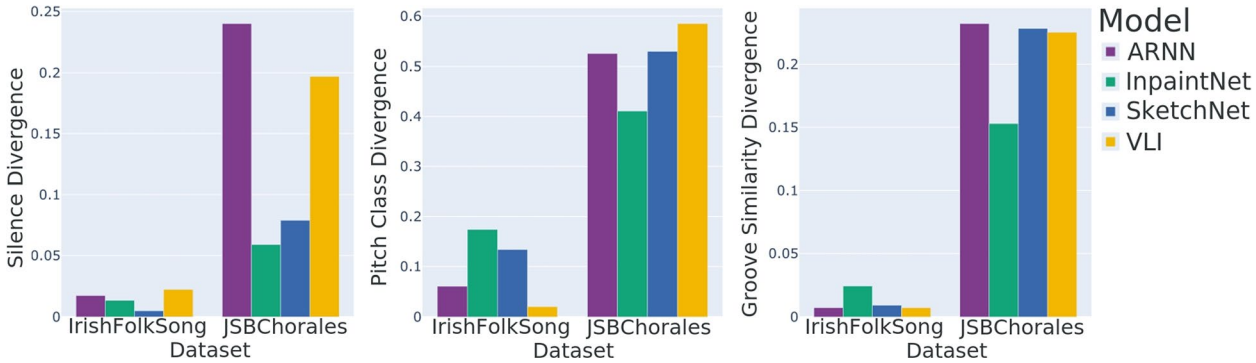


Fig. 10 Comparison of divergence metrics for IrishFolkSong and JSBChorales

3. The performance of all models varies consistently and significantly as the dataset varies.
4. When comparing performance on the note metrics (onset, pitch, and rhythm), all models rank these metrics consistently, achieving the highest results for position score, then rhythm accuracy, and finally pitch accuracy. This gives clear signals regarding which aspects of the notes are harder to capture for any model.

Considering our findings and our proposed benchmark, there are several possible extensions to this work.

For future work, we plan to extend the framework to non-single monophonic music generation techniques, especially those that have emerged after the introduction of Jukebox [5], and Diffusion Models [24], such as [6, 7]. We also consider that a combination of the features used in our metrics might be relevant to build a metric that automatically weights them in a similar way to FAD [25] metric for audio enhancement.

In terms of evaluation, additional subjective listening experiments may be helpful to exhibit correlations between our proposed metrics and human perception on the generated data. Several music generation studies have included human evaluation [14, 15, 26] supporting this idea. we are aware that a subjective evaluation study without proper guidance may struggle presenting significant scientific evidence [27], leaving this topic for future work.

The evaluation of the models could also include data augmentation strategies. Different methods such as transposing a sequence to all keys or randomly changing a note with its third or fifth may have a different impact on the models' performance.

The design of metrics can also be improved further. For instance, Pitch Accuracy currently considers all mistakes to be equally important although in practice some notes are more dissonant than others.

Similarly, new Divergence Metrics that evaluate other musical attributes could be of interest when evaluating music generation. This may include the amount of self-replication for the sequences on a given dataset, the amount of harmonies or intervals, the amount of polyphony, among others.

As a final remark, we hope that the proposed benchmark will benefit the community by paving the way to facilitate the reproducibility and evaluation of music inpainting models, as well as providing standards for the development of new ones.

Acknowledgements
We would like to thank the members of the Representations for Learning and Language group at the University of Chile and the CreativAI Lab at PUC for their valuable suggestions and comments.

Authors' contributions
Conception and design of study: MA, FB, RC, and DP; acquisition of data: MA; analysis and/or interpretation of data: MA, FB, RC, and DP. Drafting the manuscript: MA; revising the manuscript critically for important intellectual content: MA, FB, RC, and DP. Approval of the version of the manuscript to be published: MA, FB, RC, and DP.

Funding
This work has been funded by ANID - Millennium Science Initiative Program - Code ICN17_002 and the National Center for Artificial Intelligence CENIA FB210017, Basal ANID and ANID - Fondecyt projects 1230926 and 1200290.

Availability of data and materials
The code, datasets, and models utilized in this study are publicly available at https://github.com/maranedah/music_inpainting_benchmark.
Additionally, each individual dataset can be accesed directly from their corresponding source:
- JSB Chorales repository: https://github.com/cuthbertLab/music21/tree/master/music21/corpus/bach.
- AILabs1k7 repository: https://github.com/YatingMusic/compound-word-transformer/blob/main/dataset/Dataset.md.
- IrishFolkSong repository: https://github.com/IraKorshunova/folk-rnn/tree/master/data.

Declarations

Ethics approval and consent to participate
This article does not contain any studies with human participants or animals performed by any of the authors.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Received: 13 September 2022 Accepted: 14 March 2023
Published online: 05 May 2023

References

1. J.-P. Briot, G. Hadjeres, F.-D. Pachet, *Deep Learning Techniques for Music Generation*, vol. 1 (Springer, Cham, 2020)
2. A. Pati, A. Lerch, G. Hadjeres, in Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR), Delft, The Netherlands, November 4-8. Learning to Traverse Latent Spaces for Musical Score Inpainting. pp. 343–351
3. M. F. Dacrema, P. Cremonesi, D. Jannach, in Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, Copenhagen, Denmark, September 16-20, 2019. Are we really making much progress? A worrying analysis of recent neural recommendation approaches, pp. 101–109
4. A. Arango, J. Pérez, and B. Poblete, in Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019. Hate Speech Detection is Not as Easy as You May Think: A Closer Look at Model Validation, pp. 45–54
5. P. Dhariwal, H. Jun, C. Payne, J.W. Kim, A. Radford, I. Sutskever, Jukebox: A generative model for music. (2020). arXiv preprint arXiv:2005.00341
6. Y. Ren, X. Tan, T. Qin, J. Luan, Z. Zhao, and T.-Y. Liu, in KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020. DeepSinger: Singing Voice Synthesis with Data Mined From the Web, pp. 1979–1989
7. Z. Borsos, R. Marinier, D. Vincent, E. Kharitonov, O. Pietquin, M. Sharifi, O. Teboul, D. Grangier, M. Tagliasacchi, N. Zeghidour, Audiolm: a language modeling approach to audio generation. (2022). arXiv preprint arXiv:2209.03143
8. B. Bogunović, Creative cognition in composing music. New Sound Int. J. Music. **53**(1), 89–117 (2019)
9. C.-J. Chang, C.-Y. Lee, and Y.-H. Yang, in Proceedings of the 22nd International Society for Music Information Retrieval Conference, ISMIR 2021, Online, November 7-12, 2021. Variable-Length Music Score Infilling via XLNet and Musically Specialized Positional Encoding, pp. 97–104
10. K. Chen, C.-I. Wang, T. Berg-Kirkpatrick, and S. Dubnov, in Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR 2020, Montreal, Canada, October 11-16, 2020. Music SketchNet: Controllable Music Generation via Factorized Representations of Pitch and Rhythm, pp. 77–84
11. G. Hadjeres, F. Nielsen, Anticipation-rnn: Enforcing unary constraints in sequence generation, with application to interactive music generation. Neural Computing and Applications. **32**(4), 995–1005 (2020)
12. M.S. Cuthbert, C. Ariza, in *Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR 2010, Utrecht, August 9-13, 2010*, ed. by J.S. Downie, R.C. Veltkamp. Music21: A toolkit for computer-aided musicology and symbolic music data. (International Society for Music Information Retrieval, 2010), pp. 637–642. http://ismir2010.ismir.net/proceedings/ismir2010-108.pdf
13. B.L. Sturm, J.F. Santos, O. Ben-Tal, I. Korshunova, Music transcription modelling and composition using deep learning. CoRR **abs/1604.08723** (2016). http://arxiv.org/abs/1604.08723
14. G. Hadjeres, F. Pachet, F. Nielsen, in *International Conference on Machine Learning*. Sydney. Deepbach: a steerable model for bach chorales generation. (PMLR, 2017), pp. 1362–1371
15. C.-Z. A. Huang, T. Cooijmans, A. Roberts, A. C. Courville, and D. Eck, in Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017. Counterpoint by Convolution, pp. 211–218
16. D.P. Kingma, M. Welling, in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, ed. by Y. Bengio, Y. LeCun. Auto-encoding variational bayes. (2014)
17. Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R.R. Salakhutdinov, Q.V. Le, Xlnet: Generalized autoregressive pretraining for language understanding. Adv. Neural Inf. Process. Syst. **32**, 5754–5764 (2019)
18. S. Wu, Y. Yang, in *Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR 2020, Montreal, Canada, October 11-16, 2020*. ed. by J. Cumming, J.H. Lee, B. McFee, M. Schedl, J. Devaney, C. McKay, E. Zangerle, T. de Reuse. The jazz transformer on the front line: Exploring the shortcomings of ai-composed music through quantitative measures. (2020) pp. 142–149
19. M. Cartagena, Exploring symbolic music generation techniques using conditional generative adversarial networks. Master's thesis, Escuela de Ingeniería. Pontificia Universidad Católica de chile. (2021) https://repositorio.uc.cl/handle/11534/60993
20. J. Lin, Divergence measures based on the shannon entropy. IEEE Trans. Inf. Theory. **37**(1), 145–151 (1991). https://doi.org/10.1109/18.61115
21. M.M. Deza, E. Deza, Encyclopedia of distances, in *Encyclopedia of Distances*. (Springer, Cham, 2009), pp.1–583
22. F. Nielsen, On a generalization of the jensen-shannon divergence and the jensen-shannon centroid. Entropy. **22**(2), 221 (2020)
23. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. An image is worth 16x16 words: Transformers for image recognition at scale. (OpenReview.net, Austria, 2021). https://openreview.net/forum?id=YicbFdNTTy
24. J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models. Adv. Neural Inf. Process. Syst. **33**, 6840–6851 (2020)
25. K. Kilgour, M. Zuluaga, D. Roblek, and M. Sharifi, in Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019. Fréchet Audio Distance: A

Reference-Free Metric for Evaluating Music Enhancement Algorithms, pp. 2350–2354
26. F. T. Liang, M. Gotham, M. Johnson, and J. Shotton, in Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017. Automatic Stylistic Composition of Bach Chorales with Deep LSTM, pp. 449–456
27. L.-C. Yang, A. Lerch, On the evaluation of generative models in music. Neural Comput. Appl. **32**(9), 4773–4784 (2020)

Publisher's Note