SYNTHIA'S MELODY: A BENCHMARK FRAMEWORK FOR UNSUPERVISED DOMAIN ADAPTATION IN AUDIO

Chia-Hsin Lin[⋆] *Charles Jones*[⋆] *Björn W. Schuller*[⋆†] *Harry Coppock*[⋆]

[⋆]GLAM, Imperial College London, UK
[†]Chair EIHW, University of Augsburg, Germany
`cynthialin0130@gmail.com`,
`{charles.jones17, bjoern.schuller, harry.coppock}@imperial.ac.uk`

ABSTRACT

Despite significant advancements in deep learning for vision and natural language, unsupervised domain adaptation in audio remains relatively unexplored. We, in part, attribute this to the lack of an appropriate benchmark dataset. To address this gap, we present **Synthia's melody**, a novel audio data generation framework capable of simulating an infinite variety of 4-second melodies with user-specified confounding structures characterised by musical keys, timbre, and loudness. Unlike existing datasets collected under observational settings, Synthia's melody is free of unobserved biases, ensuring the reproducibility and comparability of experiments. To showcase its utility, we generate two types of distribution shifts—domain shift and sample selection bias—and evaluate the performance of acoustic deep learning models under these shifts. Our evaluations reveal that Synthia's melody provides a robust testbed for examining the susceptibility of these models to varying levels of distribution shift.

1. INTRODUCTION

While deep learning models achieve impressive performance across domains such as imaging [1], text [2], and audio [3], they are prone to learning *shortcuts* – features not representative of the intended task [4]. For example, image classifiers trained to recognise animals may instead depend on spuriously correlated background features [5], and natural language models may falsely rely on sentiment when predicting review quality [6]. Similar effects manifest across many data types and learning tasks, encompassing imaging, text [7], audio [8], and reinforcement learning [9]. This tendency poses credibility problems for deploying deep learning methods in high-stakes settings. Notably, shortcuts were leveraged heavily during the COVID-19 pandemic response to achieve falsely high diagnostic accuracy for SARSCoV2 infection using patient respiratory audio [10, 11].

Although models may not extract the intended features, shortcut learning needs not be an issue if shortcuts are present at both training and deployment time. In practice, this is unlikely to be the case [12]. We may instead view shortcut learning as part of a larger problem of *distribution shift* (or dataset shift), where the joint distribution of inputs and outputs differs between settings [13]. When building models, we usually assume that training and testing data are identically and independently distributed (*i.i.d*) [14]. However, shifts between training and testing stages are ubiquitous in practice [15]. Trained models may not be robust to dataset shifts in unseen environments.

To address the issue, research on domain adaptation methods aims to improve the robustness of models [16, 17, 18, 19]. However, most of the research focuses on the image and text domains, while relatively few address audio. We attribute this to the lack of common benchmark

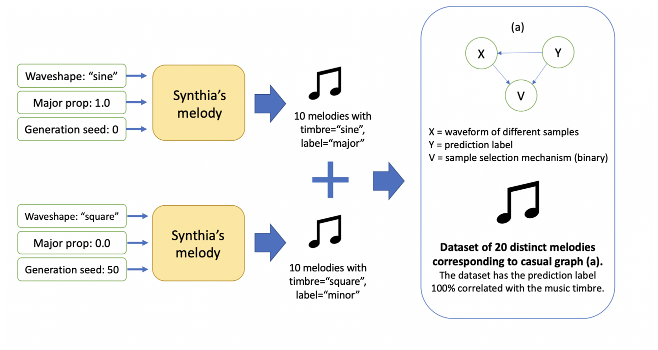

Waveshape: "sine"
Major prop: 1.0
Generation seed: 0
Synthia's melody
10 melodies with timbre="sine", label="major"
Waveshape: "square"
Major prop: 0.0
Generation seed: 50
Synthia's melody
10 melodies with timbre="square", label="minor"
(a)
X
Y
V
X = waveform of different samples
Y = prediction label
V = sample selection mechanism (binary)
Dataset of 20 distinct melodies corresponding to casual graph (a).
The dataset has the prediction label 100% correlated with the music timbre.

Fig. 1: **Example usage of Synthia's melody**. Here, we see the steps to generate a dataset of 20 samples where the music timbre and prediction label are 100% correlated. The generated data has a causal structure represented by the causal graph (a). We point the user to the GitHub `README.md` file for a full set of instructions and use cases.

datasets in audio akin to coloured-MNIST [12] in vision and MNLI [20] in texts. In audio, [21] inject biases into observational speech datasets with five data augmentation methods: mp3 compression, additive white noise, loudness normalisation, non-speech zeroing, and μ-law encoding. Although the results show that models are prone to the injected shortcuts, unobserved artefacts in the original speech data, such as microphone mismatch, could still alter the experimental outcomes. Moreover, the research only considers cases where training data is 100% or 0% perturbed, which limits the ability to observe model behaviours with varying shift levels.

To foster the corresponding research in audio, we propose a fully synthesised data generation mechanism called **Synthia's melody**. The mechanism generates 4-second melody samples in the form of `.wav` file at a sample rate of 16 000 Hz for a music key (binary) classification task. The simulated data exhibit several attractive properties. First, it ensures the reproducibility of experiment results, as the data depend only on pre-written scripts in contrast to most audio data collected under observational settings. Second, researchers can generate desired shifts by modifying the mechanism parameters. Third, the generated data are interpretable by humans: researchers can hear the generated melody and feel differences if shifts occur. The example usage is shown in Figure 1. Code and audio samples are available at `https://github.com/cynthpie/Synthia_melody`.

2. BACKGROUND

Domain adaptation Let $\mathcal{X}$ and $\mathcal{Y}$ denote the input and label space, respectively. We denote the domain D with $D = \{x^{(i)}, y^{(i)}\}_{i=1}^{n} \sim p^D(x, y)$, where $x \in \mathcal{X}, y \in \mathcal{Y}$, and $p^D(x, y)$ is the joint distribution of the corresponding random variables X, Y that generates D. Given that dataset shift exists, the goal of domain adaptation methods is to learn a predictive function $h : \mathcal{X} \to \mathcal{Y}$ from the source (training) domain D_{train} that minimises the predictive risk R in a similar but unseen domain D_{test}, given that $p^{test}(x, y) \neq p^{train}(x, y)$. The predictive risk function can be written as

$$R = \min_h \mathbb{E}_{(x,y)\in D^{test}}[\ell(h(x), y)], \tag{1}$$

where $\mathbb{E}$ is the expectation and $\ell(\cdot, \cdot)$ is the loss function [1].

Related music theory We set the generated data in the context of music. Readers unfamiliar with music theory are referred to [22, 23]. A melody is composed of a sequence of musical tones. Each musical tone has four auditory attributes: pitch, duration, timbre, and loudness. In signal processing, such attributes correspond to frequency, time, waveshape, and amplitude. In Western music, there are 24 music keys; half are major, and the other half are minor. For most people, major keys sound happy and minor keys sound sad. Each music key has seven notes in its corresponding scale. The set of 7 notes is distinct in each of the 24 keys. Certain combinations of the seven notes form chords. Common chords include triads and sevenths, where three notes form triads and sevenths are formed by four. Roman numerals often denote chords. For example, the first triad and fifth seventh in a major key are denoted as I and V$_7$, respectively. We say a melody is in a key if it is composed by the chords in that key.

3. METHOD

To simulate melodies, we need to define four auditory attributes of each musical tone: pitch, duration, timbre, and loudness.

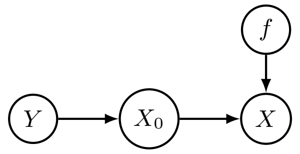
Oscillator and ADSR envelope We determine timbre and loudness with two components: oscillator and ADSR envelope. An oscillator generates waves with a given frequency (pitch) and amplitude (loudness). Oscillators with different waveshapes create different timbres. We use sine, square, sawtooth, and triangle oscillators in this study. The ADSR envelope alters the amplitude of waves that oscillators generate. The amplitude change in a melody can be "stable", "increase", or "decrease", where details can be found in the Appendix, amplitude change.

Melody generation Given timbre and loudness defined, we define pitch and duration, or melodies, by using random sampling algorithms. To generate a melody, we randomly draw a music key K among 12 major/minor keys given a label $Y \in \{\text{major}, \text{minor}\}$. We then randomly draw N chords $C_{i=1}^{N}$ in the key K, where N is uniformly sampled from some integer set. We consider 10 chords: $C_{\text{major}} = \{$I ii iii IV V vi vii$^{\text{o}}$ ii$_7$, V$_7$, vii$^{ø}{}_7\}$ and $C_{\text{minor}} = \{$i ii$^{\text{o}}$ III^{+} iv V VI vii$^{\text{o}}$ ii$^{ø}{}_7$, V$_7$, vii$^{\text{o}}{}_7\}$ for major and minor samples, respectively. For each chord, we sample their duration $T_{i=1}^{N}$ independently from some continuous distribution. If $\sum T_{i=1}^{N}$ is less than 4 seconds, we repeat the melody until the targeted duration is reached. The melody generation algorithm is detailed in the Appendix, algorithm 1.

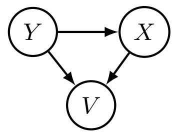
[1]The equation for R represents the theoretical risk we aim to minimise in the target domain D_{test}. It is important to note that we cannot directly compute this risk during training given the unsupervised domain adaptation setting.

Data generation We generate 50 000 melodies with random seeds from 0 to 49 999 and perform train-val split to obtain 40 000 training and 10 000 validation data. We generate 10 000 melodies with random seeds from 55 000 to 64 999. The process is repeated for four timbres: sine, square, sawtooth, and triangle. We fix the amplitude to "stable" for all samples[2]. As such, the training, validation, and test sets of the four timbres are acoustically indistinguishable except for their timbres.

Shifts considered We represent distribution shifts with causal graphs [24, 25], where the shifts are treated as outcomes of interventions. We focus on anticausal tasks [26], where the input X is caused by the label Y, such that $Y \to X$. We consider two types of shifts: domain shift and sample selection bias [13] detailed in Figure 2.



(a) **Domain Shift**. The observed covariate X is affected by some mapping f which varies during training and testing. The goal of the classifier is to learn $p(y|x_0)$ via X, given that X_0 is unobserved.



(b) **Sample Selection Bias**. The selection process $V \in \{0, 1\}$ depends on both the input X and label Y. The dependency of V on X and Y varies between training and testing time.

Fig. 2: The two types of shift considered in this study.

Domain shift (Figure 2a) refers to cases when the observed covariate X is affected by some mapping f, which varies across training and testing. Given varying f, the learnt distribution $p^{train}(y|x)$ has no guarantee to be the same as $p^{test}(y|x)$. The goal of a conditional classifier is to learn the domain-invariant conditional distribution $p(y|x_0)$ via X given X_0 unobserved.

Sample selection bias (Figure 2b) refers to cases when the sample section process V depends on both X and Y, and the dependency varies between training and testing time. We denote the selection process with a binary variable V, where the event $V = 1$ indicates that the sample is being selected and $V = 0$, otherwise. Given the varying dependency of V, the learnt distribution in training $p^{train}(y|x)$ may not be applicable to that of testing time $p^{test}(y|x)$.

Shift construction We use timbre to construct two types of dataset shifts. For domain shift, we represent two domains by samples generated by sine and square waves. We consider 12 shift levels by gradually replacing the sine sample with the square ones in the training data until the number of sine and square samples are equal. For sample selection bias, we construct a biased sample by correlating the timbre with the prediction label. Specifically, we generate major samples with sine waves and minor ones with square waves. We consider 11 shift levels by varying the proportion of biased samples in the training data. We evaluate trained models on three test sets: in-distribution, neutral, and anti-bias sets. The in-distribution set is the test set that has the same proportion of biased sample as the training set. The neutral set has no correlation between the timbre

[2]experimenting with using different amplitudes, e. g., "increase" or "decrease" or a custom config left for future work.

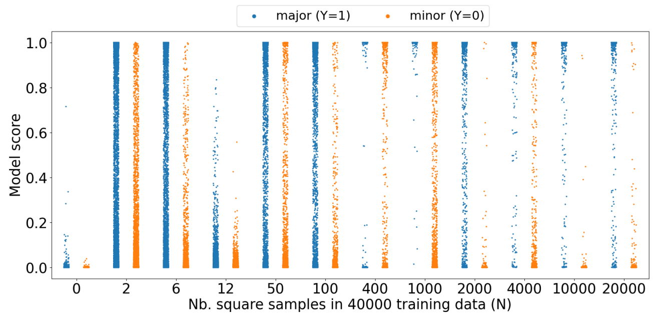


Fig. 3: **Distribution of model score of baseline models on the square test set with varying number of square samples (N) in the training data**. Each point represents the sigmoid prediction of a square test sample. Colours represent the music key, where major samples are expected to have scores close to 1 and minors with 0.

and the prediction label. The anti-bias set has the same proportion of the biased samples as the training set, while the bias is reverted, e. g. major samples in square waves and minor ones in sine waves. The three test sets are built by extracting the required samples from the sine and square test sets to maintain low compute and time costs.

Model We consider two models: baseline and Domain-Adversarial Neural Network (DANN). The baseline model is a SampleCNN[27] with 9 Res-2 blocks[28]. We set all kernel sizes, max-pooling sizes to 3 and stride sizes to 1 in all convolutional layers. The DANN model is a SampleCNN trained with the domain adversarial training algorithm developed by [16]. A DANN consists of a feature encoder, a classifier, and a domain discriminator. The domain discriminator distinguishes samples in sine and square waves. To make the results of the two models comparable, we use 1 Res-2 block in the feature encoder, 8 Res-2 blocks in the label classifier, and 2 Res-2 blocks in the domain discriminator. In this way, the total parameter sizes of the feature encoder and classifier are the same as the baseline model.

4. RESULT

The main purpose of Synthia's melody is to provide a tool researchers can use to evaluate audio-based machine learning models in different dataset shift settings. To demonstrate Synthia's melody's utility, we evaluate the baseline and DANN models across varying levels of shift for a range of different shifts.

Domain Shift Figure 3 details the logit output of the baseline model when evaluated on varying levels of domain shift. Here, we see the model performs poorly when no square samples appear in the training set, predicting all square samples as minor. Interestingly, we see a drastic change in behaviour when just two square samples are injected into the training, with the spread of logit scores becoming considerably broader. Inspecting Figure 4, which details the corresponding accuracy scores, we see that, despite the behaviour change in logit output, a considerable proportion of square samples are needed before model performance approaches that of an in-distribution test set. Here, we demonstrate how Synthia's Melody can be used to uncover interesting relationships between domain shift, model architecture, and performance.

Sample Selection Bias Figure 6 shows the logit outputs of baseline models on the neutral test set with varying levels of sample selection bias in training data. We examine the model score from two aspects: key (Figure 6a) and the confounding wave shape (Figure 6c). At shift levels greater than 0.6, less confident predictions are made, first with major and then with minor classes. When the shift strength is the largest (shift level=1.0), the models do not learn the target task at all, making bias-leading predictions based on wave shapes only – see Figure 6c.

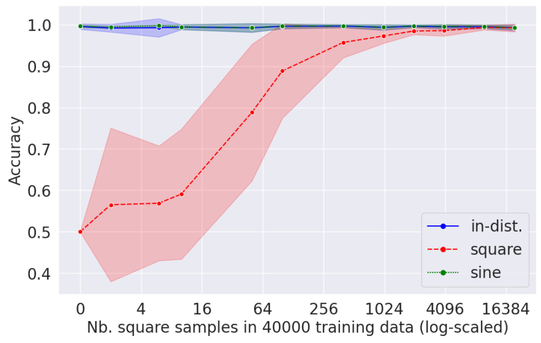


Fig. 4: **Test accuracy of baseline models trained on datasets with 12 levels of domain shift**. The experiment is repeated five times. The line represents the sample mean of the five test accuracies. The 95% error bands are calculated with ± 2 standard deviations from the sample mean, assuming that the test accuracy follows a normal distribution.

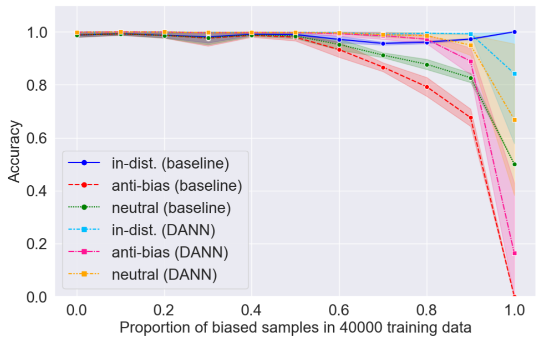
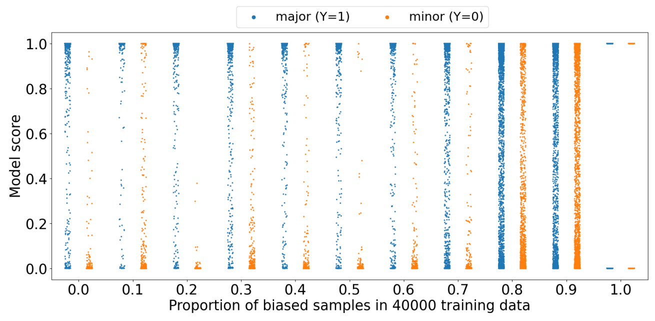
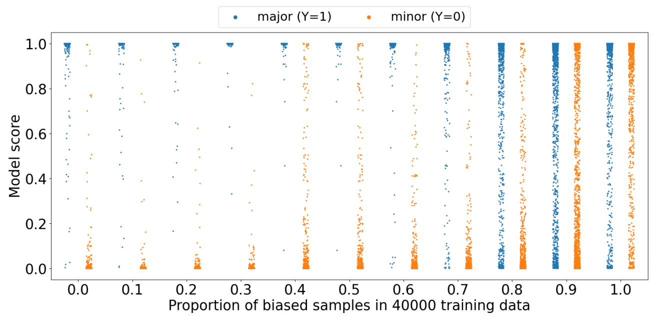


Fig. 5: **Comparison between baseline and DANN models test accuracy across 11 levels of sample selection bias.** The experiment is repeated five times for baseline models and three times for DANN. The line represents the sample mean of the five test accuracies. The bands are calculated with ± 1 standard deviation from the sample mean, assuming that the test accuracy follows a normal distribution.

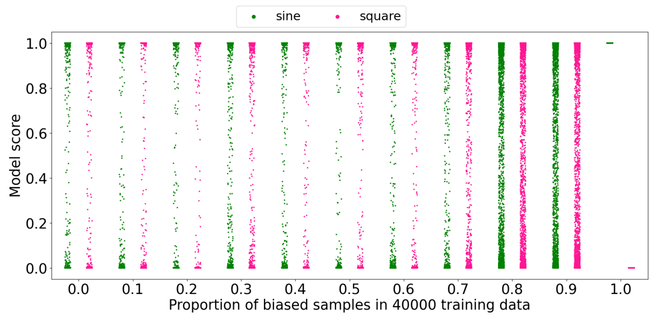
Figure 6b shows the DANN model scores with varying shift levels of sample selection bias injected in training data. The result suggests that DANN is more resistant than the baseline, as scores remain polarised at higher levels of shift than in Figure 6a. It is important to note that at extreme levels of shift, DANN loses its robustness, evident in the incorrect prediction of 1 for all sine samples in Figure 6d. Figure 5 compares the test accuracy of the baseline and DANN models at varying levels of sample selection bias. For the baseline model, the three test accuracies diverge at a shift level of 0.7, which is reflected in Figure 6a. We see that as the shift level increases, the model accuracy on the in-distribution set remains high, but the



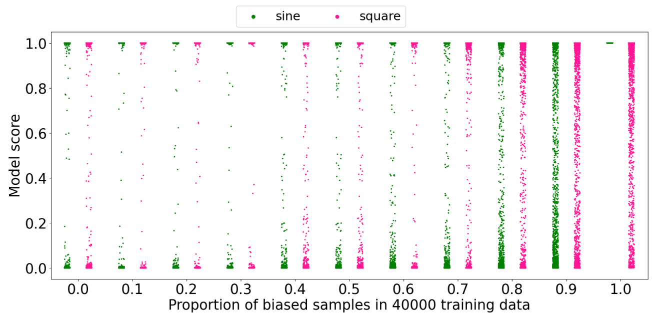
(a) Baseline model scores on the neutral test set with varying proportions of biased sample (shift level) in training data.



(b) DANN model scores on the neutral test set with varying proportions of biased sample (shift level) in training data.



(c) Baseline model scores with wave shape indication.



(d) DANN model scores with wave shape indication.

Fig. 6: **Comparison of Model and DANN model scores on the neutral test set**. (a) and (c) show the scores of the baseline model, while (b) and (d) show the scores of the DANN model.

accuracy on neutral and anti-bias sets starts to drop. Specifically, when the shift level equals 1.0, the anti-bias accuracy becomes 0.0, suggesting the baseline model does not learn any signals except the biases. Here, the shortcut is so strong that it acts as a mask, preventing the model from learning any other features despite having ample capacity. This is corroborated by the no-better-than-random neutral test set performance.

Figure 7 shows the baseline model and DANN test accuracy on melodies with unseen wave shapes during training (sawtooth and triangle) as the level of sample selection bias (sine and square) increases in the training set. As shown in Figure 5, the test accuracy drops as the shift level increases, identifying a reliance on leveraging sine vs square timbre for key prediction. The result shows that DANN is more robust and generalises to unseen wave shapes better than the baseline model. Interestingly, the DANN trained on data with a shift level of 0.4 performs better on unseen wave shapes than ones trained with a lower shift level, such as 0.0 and 0.1. This suggests that injecting a small amount of bias in the training data may actually boost the DANN model's robustness. We hypothesise that a mild amount of biases increases the DANN model's incentive to learn features invariant to the bias feature as the reverted gradients from the discriminator will get larger. There is also the factor of the discriminator acting as a regulariser, injecting noise into the system, through the inverted gradients.

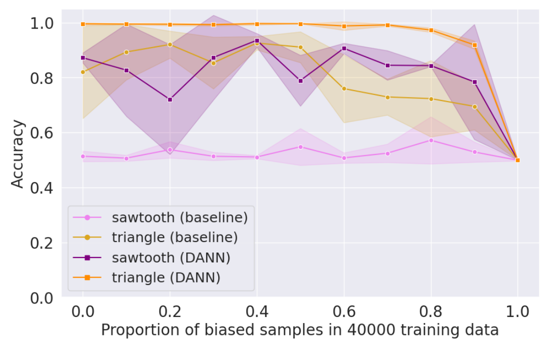


Fig. 7: **Test accuracy of DANN on samples with unseen wave shapes during training**. The x-axis represents the proportion of the biased samples in the sine/square training data. The experiment is repeated five times for baseline models and three times for DANN. The line represents the sample mean and the error bands are calculated with $\pm$ 1 standard deviation from the sample mean.

5. CONCLUSION

We presented Synthia's melody, a robust framework for examining the susceptibility of deep audio models to distribution shifts. All melody samples are free of unobserved biases, given their synthetic nature. We detailed novel model behaviour under varying levels of shifts constructed via music timbre. We considered two distribution shifts, domain shift and sample selection bias, and two types of models, a SampleCNN baseline and DANN, where DANN is a SampleCNN trained with a domain-adaptation algorithm. In two types of shift, we showed baseline models make more confident predictions with lower shift levels and notably, in sample selection bias, we showed the injected bias stops the baseline models from learning the target task even if they have enough capacity to do so. The evaluation demonstrates that DANN is more robust to the injected bias and can boost the model's robustness up to, but not including, extreme levels of shift. Synthia's melody provides a robust testbed that allows for reproducible results and serves as an important evaluation framework for the development of future domain adaptation algorithms. Moving forward, the insights gained from Synthia's melody offer vital avenues for enhancing the resilience of deep audio models.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[3] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold *et al.*, "Cnn architectures for large-scale audio classification," in *2017 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, 2017, pp. 131–135.

[4] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann, "Shortcut learning in deep neural networks," *Nature Machine Intelligence*, vol. 2, no. 11, pp. 665–673, nov 2020. [Online]. Available: https://doi.org/10.1038%2Fs42256-020-00257-z

[5] S. Beery, G. Van Horn, and P. Perona, "Recognition in terra incognita," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[6] V. Veitch, A. D' Amour, S. Yadlowsky, and J. Eisenstein, "Counterfactual Invariance to Spurious Correlations in Text Classification," in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 16 196–16 208.

[7] T. Niven and H.-Y. Kao, "Probing neural network comprehension of natural language arguments," 2019.

[8] B. Chettri, "The clever hans effect in voice spoofing detection," in *2022 IEEE Spoken Language Technology Workshop (SLT)*, 2023, pp. 577–584.

[9] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R. Müller, "Unmasking clever hans predictors and assessing what machines really learn," *Nature communications*, vol. 10, no. 1, p. 1096, 2019.

[10] H. Coppock, G. Nicholson, I. Kiskin, V. Koutra, K. Baker, J. Budd, R. Payne, E. Karoune, D. Hurley, A. Titcomb, S. Egglestone, A. T. Cañadas, L. Butler, R. Jersakova, J. Mellor, S. Patel, T. Thornley, P. Diggle, S. Richardson, J. Packham, B. W. Schuller, D. Pigoli, S. Gilmour, S. Roberts, and C. Holmes, "Audio-based ai classifiers show no evidence of improved covid-19 screening over simple symptoms checkers," 2023.

[11] H. Coppock, L. Jones, I. Kiskin, and B. Schuller, "Covid-19 detection from audio: seven grains of salt," *The Lancet Digital Health*, vol. 3, no. 9, pp. e537–e538, 2021.

[12] M. Arjovsky, L. Bottou, I. Gulrajani, and D. Lopez-Paz, "Invariant risk minimization," *arXiv preprint arXiv:1907.02893*, 2019.

[13] J. Q. Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, "Dataset shift in machine learning," *The MIT Press*, vol. 1, p. 5, 2009.

[14] V. Vapnik, "An overview of statistical learning theory," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 988–999, Sep. 1999.

[15] M. A. Alcorn, Q. Li, Z. Gong, C. Wang, L. Mai, W.-S. Ku, and A. Nguyen, "Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4845–4854.

[16] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," 2016.

[17] B. Kim, H. Kim, K. Kim, S. Kim, and J. Kim, "Learning not to learn: Training deep neural networks with biased data," 2019.

[18] T. Schuster, D. J. Shah, Y. J. S. Yeo, D. Filizzola, E. Santus, and R. Barzilay, "Towards debiasing fact verification models," 2019.

[19] J. Wang, C. Lan, C. Liu, Y. Ouyang, T. Qin, W. Lu, Y. Chen, W. Zeng, and P. Yu, "Generalizing to unseen domains: A survey on domain generalization," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[20] A. Williams, N. Nangia, and S. Bowman, "A broad-coverage challenge corpus for sentence understanding through inference," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 1112–1122. [Online]. Available: https://aclanthology.org/N18-1101

[21] H. jin Shim, R. Gonzalez Hautamäki, M. Sahidullah, and T. Kinnunen, "How to Construct Perfect and Worse-than-Coin-Flip Spoofing Countermeasures: A Word of Warning on Shortcut Learning," in *Proc. INTERSPEECH 2023*, 2023, pp. 785–789.

[22] J. G. Roederer, *The physics and psychophysics of music: An introduction*. Springer, 2008, vol. 4.

[23] C. Schmidt-Jones, "Understanding basic music theory," 2013.

[24] J. Pearl, *Causality*. Cambridge university press, 2009.

[25] D. C. Castro, I. Walker, and B. Glocker, "Causality matters in medical imaging," *Nature Communications*, vol. 11, no. 1, p. 3673, 2020.

[26] B. Schölkopf, D. Janzing, J. Peters, E. Sgouritsa, K. Zhang, and J. Mooij, "On causal and anticausal learning," *arXiv preprint arXiv:1206.6471*, 2012.

[27] J. Lee, J. Park, K. L. Kim, and J. Nam, "Samplecnn: End-to-end deep convolutional neural networks using very small filters for music classification," *Applied Sciences*, vol. 8, no. 1, p. 150, 2018.

[28] T. Kim, J. Lee, and J. Nam, "Comparison and analysis of samplecnn architectures for audio classification," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 2, pp. 285–297, 2019.

6. APPENDIX

Tuning We tune all pitches to concert pitch with the reference pitch A4 equals to 440 Hz. Table 1 show pitches in the fourth octave and their corresponding frequencies. Pitches in other octaves can be extended by doubling or halving the frequency of the corresponding pitch, e. g., A3=220 Hz and A5=880 Hz.

Pitch	Frequency (Hz)
C4	261.6256
D$^\flat$4	277.1826
D4	293.6648
E$^\flat$4	311.1270
E4	329.6276
F4	349.2282
G$^\flat$4	369.9944
G4	391.9954
A$^\flat$4	415.3047
A4	440.0000
B$^\flat$4	466.1638
B4	493.8833

Table 1: **Pitches used in Synthia's melody**. This table shows the frequency of the pitches in the 4th octave. Pitches in other octaves can easily be extended by doubling or halving the frequency of the corresponding pitch, e. g., A3=220 Hz and A5=880 Hz.

Waveshape We consider four types of waveshapes in this study: sine, square, sawtooth, and triangle. The waveshapes are illustrated in Figure 8. Different wave shapes result in different music timbres. Demos are available at `https://drive.google.com/drive/folders/13PLu_ZZ7rv9vi5pZWapqebLambOjAElB`.

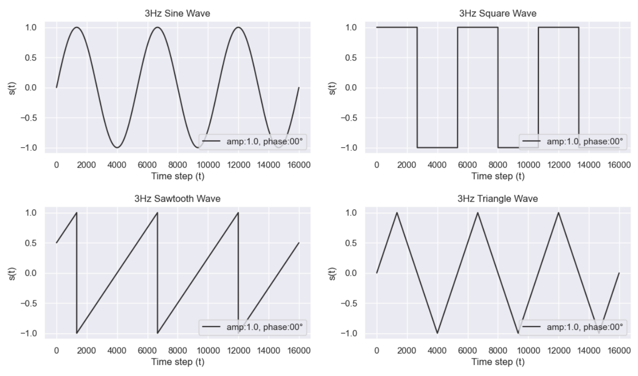


Fig. 8: **Four types of wave shapes**. Each of the waves has a fundamental frequency=3 Hz, amplitude=1.0, and phase=0.0.

Amplitude change We use ADSR envelopes to alter the amplitude in melody. An ADSR envelope is defined by the four parameters: attack (A), decay (D), sustain (S), and release (R). We set the amplitude change "increase" with A=2, D=0.01, S=1, R=0.01; "decrease" with A=0.01, D=0.01, S=1, D=2, and "stable" with A=0.01, D=0.01, S=1, D=0.01, where the unit of A, D, R are in seconds, and $S = 1$ indicates the maximum volume.

Chords considered We use major and harmonic minor scales to construct major and minor keys. The corresponding seven basic triads are listed in Table 2. In addition to triads, we also considered the sevenths $\{\text{ii}_7\ \text{V}_7\ \text{vii}^{ø}{}_7\}$ for major keys and $\{\text{ii}^{ø}{}_7\ \text{V}_7\ \text{vii}^{o}{}_7\}$ for minor keys.

Triads	1	2	3	4	5	6	7
Major key	I	ii	iii	IV	V	vi	vii^{o}
Minor key	i	ii^{o}	III^{+}	iv	V	VI	vii^{o}

Table 2: **Triads considered in major and minor keys**. We denote triads by Roman numerals. The quality of chords are denoted as major (uppercase), minor (lowercase), augmented (+), and diminished (o).

Melody generation algorithm Algorithm 1 shows the melody generation process used in Synthia's melody. The algorithm can be broken down into eight stages: 1. Draw a music key K (line 1); 2. Construct the scale corresponding to K given the key type $Y \in \{\text{major, minor}\}$ (line 2); 3. Determine the number of chords present in the melody, denoted as N (line 3); 3. Draw N triads $C_{i=1}^{N}$ from the key Y (line 4); 4. Ensure triads I, IV, V (i iv V for minor keys) are present (line 6 to 30); 5. Change certain triads to their corresponding sevenths with 0.5 probability (line 31 to 34); 6. Randomly alter the octave of each note in each chord (line 36 to 41); 7. Assign a duration T to each chord to form a melody (line 42 to 44); 8. If the total duration does not reach 4 seconds, repeat the melody until the targeted time is reached (line 45 to 49). In this study, we set $R_f = [130.81, 523.25]$ Hz, i.e. pitches ranged from C2 to C5, $R_n = \{3, 4, \ldots, 7\}$, and $R_t = [0.2, 0.9]$ seconds.

Human evaluation We randomly select 10 samples generated by Algorithm 1 and ask the general audience to perform the music key classification task. We provide one melody demo in a major key and the other one in a minor key at the beginning of the questionnaire for those without professional musical training. The questionnaire is available at `https://forms.gle/pan5kaMRxREXtBNK6`. The result is shown in figure 9. In the questionnaire, we also asked the participants to provide their professional musical experience in years, and find no significant correlation between higher scores and the musical experience. For example, two of the participants who scored a 10 have no professional musical experience.

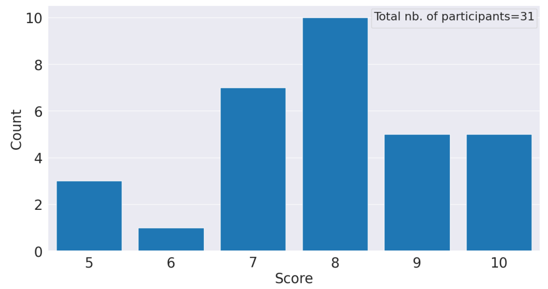


Fig. 9: **Distribution of human evaluation score on 10 melody samples generated by Algorithm 1**. The x-axis represents the number of correctly identified samples out of 10 samples. The y-axis represents the number of participants who achieved the score. The 31 participants achieved an averaged and a median score of 8.

Algorithm 1 Melody generation process

```
Require: Frequency range $R_f = [f_{\text{min}}, f_{\text{max}}]$, key type $Y \in \{\text{major}, \text{minor}\}$, number of chords set $R_n = \{n_{\text{lower}}, \dots, n_{\text{upper}}\}$, Duration range $R_t = [t_{\text{lower}}, t_{\text{upper}}]$
Ensure: One 4-second melody sample
 1: Draw a frequency $f_1$ from the 12 frequencies in table 1
 2: Construct a set of pitches $S = \langle f_1, f_2, \dots, f_7 \rangle$ given $f_1$ and $Y$
 3: Sample $N$ uniformly from $R_n$
 4: Sample $N$ triads from Table 2 uniformly with replacement given $Y$ to form a set of chord types $S_{\text{type}} = \langle C^{(1)}_{\text{type}}, C^{(2)}_{\text{type}}, \dots, C^{(N)}_{\text{type}} \rangle$, where each $C^{(i)}_{\text{type}}$ is a triad
 5: Construct an integer set $L = \{1, 2, \dots, N\}$.
 6: if $Y =$ major then
 7:   while triad I and IV and V not in $S_{\text{type}}$ do
 8:     Sample $l$ uniformly from $L$
 9:     if triad I not in $S_{\text{type}}$ then
10:       replace $C^{(l)}_{\text{type}}$ with triad I
11:     else if triad IVnot in $S_{\text{type}}$ then
12:       replace $C^{(l)}_{\text{type}}$ with triad IV
13:     else if triad V not in $S_{\text{type}}$ then
14:       replace $C^{(l)}_{\text{type}}$ with triad V
15:     end if
16:     Remove $l$ from $L$
17:   end while
18: else
19:   while triad i and iv and V not in $S_{\text{type}}$ do
20:     Sample $l$ uniformly from $L$
21:     if triad inot in $S_{\text{type}}$ then
22:       replace $C^{(l)}_{\text{type}}$ with triad i
23:     else if triad iv not in $S_{\text{type}}$ then
24:       replace $C^{(l)}_{\text{type}}$ with triad iv
25:     else if triad V not in $S_{\text{type}}$ then
26:       replace $C^{(l)}_{\text{type}}$ with triad V
27:     end if
28:     Remove $l$ from $L$
29:   end while
30: end if
31: Sample $coin$ uniformly from $\{0, 1\}$
32: if $coin$=1 then
33:   Replace all triads ii, V, vii$^{\text{o}}$ in $S_{\text{type}}$ with ii$_7$, V$_7$, vii$^{\text{ø}}{}_7$ if $Y =$ major or ii$^{\text{ø}}{}_7$, V$_7$, vii$^{\text{o}}{}_7$ if $Y =$ minor
34: end if
35: for $i = 1$ to $N$ do
36:   Construct chord $C_i = \langle p_1, p_2, p_3 \rangle$ if triad and $\langle p_1, p_2, p_3, p_4 \rangle$ if seventh corresponds to $C^{(i)}_{\text{type}}$, where each $p$ is a frequency in $S$
37:   for j = 1 to length($C_i$) do
38:     Construct a set of integer multiples of $p_j$ in $C_i$, denoted as $O_j = \{\dots, p_j/4, p_j/2, p_j, 2p_j, 4p_j, 8p_j \dots\} \in R_f$.
39:     Sample $o_j$ uniformly from $O_j$
40:     Replace $p_j$ with $o_j$
41:   end for
42:   Sample $T_i \sim U(t_{\text{lower}}, t_{\text{upper}})$
43:   Assign time $T_i$ to chord $C_i = \langle o_1, o_2, o_3 \rangle$ if triad or $\langle o_1, o_2, o_3, o_4 \rangle$ if seventh
44: end for
45: Form a melody $m = \langle C_1, C_2, \dots, C_N \rangle$
46: Calculate $T_m = \sum_{i=1}^{N} T_i$
47: while $T_m < 4$ seconds do
48:   Repeat the melody $m$ until $T_m \geq 4$
49: end while
```