REAL-TIME PERCUSSIVE TECHNIQUE RECOGNITION AND EMBEDDING LEARNING FOR THE ACOUSTIC GUITAR

Andrea Martelloni
Queen Mary University of London
a.martelloni@qmul.ac.uk

Andrew P McPherson
Imperial College
andrew.mcpherson@imperial.ac.uk

Mathieu Barthet
Queen Mary University of London
m.barthet@qmul.ac.uk

ABSTRACT

Real-time music information retrieval (RT-MIR) has much potential to augment the capabilities of traditional acoustic instruments. We develop RT-MIR techniques aimed at augmenting percussive fingerstyle, which blends acoustic guitar playing with guitar body percussion. We formulate several design objectives for RT-MIR systems for augmented instrument performance: (i) causal constraint, (ii) perceptually negligible action-to-sound latency, (iii) control intimacy support, (iv) synthesis control support. We present and evaluate real-time guitar body percussion recognition and embedding learning techniques based on convolutional neural networks (CNNs) and CNNs jointly trained with variational autoencoders (VAEs). We introduce a taxonomy of guitar body percussion based on hand part and location. We follow a cross-dataset evaluation approach by collecting three datasets labelled according to the taxonomy. The embedding quality of the models is assessed using KL-Divergence across distributions corresponding to different taxonomic classes. Results indicate that the networks are strong classifiers especially in a simplified 2-class recognition task, and the VAEs yield improved class separation compared to CNNs as evidenced by increased KL-Divergence across distributions. We argue that the VAE embedding quality could support control intimacy and rich interaction when the latent space's parameters are used to control an external synthesis engine. Further design challenges around generalisation to different datasets have been identified.

1. INTRODUCTION

There is increasing interest in deep neural networks for processing audio in real time with sufficiently low latency to be used in musical performance. There is also a drive to provide small self-contained platforms that could perform inference *at the edge*, that is, on a device that can be fitted in a musical interface or a musical instrument [1–3]. Many of the tasks in Music Information Retrieval, such as onset detection [4], playing technique classification [5], timbre transfer [6], re-synthesis of musical information [7] and generative composition [8], find an application in the design of Digital Musical Instruments (DMI) and augmented instruments, as long as the solutions conform to real-time requirements. For Real-Time MIR (RT-MIR), two physical constraints that limit the application of Deep Neural Network (DNN) models are *causality*, implying the inability to look into the future, and *low action-to-sound latency* [9]. Acceptable action-to-sound latency in music performance was found to be 10 ms [10] for percussion instruments, and the latency's *jitter* (the variation) was also found to be a factor in the quality of the interaction [11]. Although there are ways to work around higher latencies, for example by synthesising generic attacks before a specific sound is generated [12], the ideal approach would be to develop a system fulfilling the latency constraints in the first place.

In this work, we investigate RT-MIR for the processing and mapping of guitar body hit sounds to augment the timbral palette of the instrument in percussive fingerstyle. Percussive fingerstyle is an extended guitar technique that uses layered arrangements, alternate tunings and hits on the guitar's body to create the impression of a "one-man band" [13]. Our method relies on deep learning to develop recognition and embedding learning of guitar body percussion. Our model addresses the task of generating representations of body hits according to performers' percussive gestures, separating them by hand part and location. One possible application is to map such a description as parameters for a synthesis engine, such as real-time physics-based synthesis. We adapted an Automatic Drum Transcription (ADT) model based on a Convolutional Neural Network (CNN) to fit the practical constraints of an augmented instrument for percussion. Our longer-term aim is to design a network that not only works as a classifier, but also describes guitar body hits with a set of features unique for each sample, to support control intimacy [14] and try to achieve the same level of nuance afforded by acoustic instruments. To this end, we propose a variation of our model that jointly trains a classifier and a Variational Autoencoder (VAE) [15].

2. BACKGROUND

Percussion DMIs. In opposition to the direct control offered by acoustic percussion, digital percussion instru-

ments have historically afforded indirect control of discrete events [16], with hit dynamics often being the only expressive parameter over individual hits. Jathal [17] provides a detailed description of commercial digital percussion instruments, emphasising the fact that they force the player to adapt their technique to the tool, usually a set of buttons or a zone-based sample trigger. The author also advocates for the design of interfaces that interpret the technique that performers of a particular acoustic instrument have already mastered: traditional techniques will be the first sensorimotor reference that expert players will use for the exploration of DMIs [18], and they have been used in the past as the basis for controllers to navigate synthesiser spaces [19].

Hit classification. One approach is to take data from audio transducers or other sensors for on-the-fly event detection and classification, and the use of that data to trigger the generation of a sound associated to that category. Examples are Turchet *et al.*'s Smart Cajón [20], Jathal's HandSolo [17] and Zamborlin *et al.*'s Mogees [21]. This approach is well supported by music tools and software for machine learning in music such as `bonk~` for Max/PD [22], timbreID's `barkSpec~` and the Wekinator [23]. This has also been applied to the acoustic guitar through the work of Lähdeoja [24] and Stefani *et al.* [3, 25], the latter applying fully-connected DNN layers for multi-class classification of guitar techniques, including percussive ones. No direct attempts have been made to use machine learning to achieve a description beyond classification, especially one that would support Moore's *control intimacy* [14].

Automatic Drum Transcription in MIR. A task related to guitar percussion classification in MIR is Automatic Drum Transcription, the audio-based detection and inference of score notations for percussive parts. Current literature does not only address the Western drum kit, but also other percussion instruments such as the tabla [26]. A recent review [27] reports that, in the current state of the art in ADT, solutions either use non-negative matrix factorisation (NMF) or look into the relationships between hits and tackle the problem with a language model or a recurrent neural network. The most relevant system for our application is based on a CNN that jointly performs event detection and classification for ADT using a sliding buffer of 150 ms [28] and was trained on the MIREX17 drums dataset [29]. Mattur Ananthanarayana (MA) *et al.* fine-tuned the model on a dataset of tabla strokes, noting a resemblance between Kick Drum, Snare Drum and Hi-Hat sounds and the tabla strokes themselves [30].

Real-time DNNs for music performance. Our purpose is not the direct re-synthesis of guitar body hits, but rather the control of the parameters of a synthesis engine. However, we are inspired by the introduction of neural networks as tools for music performance, through solutions such as Neural Audio Synthesis and Neural or Differentiable Digital Signal Processing (DDSP). Bottlenecks and latent spaces have been used with VAEs [31, 32] and autoencoder-like structures [33] for re-synthesis of sounds and timbre transfer, with successful real-time implementations such as RAVE [34] and DDX7 [7]. NN-based solutions have also been applied to model linear [35] and non-linear audio systems, such as guitar amplifiers [36] and stomp-box overdrives [37, 38]. Solutions exist to load an arbitrary neural DSP network into a plugin to be run in a DAW, such as the Neutone VST host by Qosmo [1] and IRCAM's `nn~` [2] Max/PD external.

Tool	Latency
`bonk~` (Puckette [22])	6 ms
Stefani *et al.* [3]	20 ms
RAVE (Caillon *et al.* [34])	DAW-defined
Mogees (Zamborlin [21])	23 ms
HandSolo (Jathal [17])	17 ms
Tabla stroke classifier (MA *et al.* [30])	150 ms

Table 1: Reported buffer values for detection and inference for some of the works and studies cited in this section.

Latency. The impact of latency and jitter in music performance systems was investigated, for example, by McPherson *et al.* [11]. Table 1 reports the measurements published by the authors cited so far on the latency of their tools, specifically the duration of analysis and inference rather than audio input-to-output latency, which is system-dependent more than algorithm-dependent. Most tools that are meant for real-time use achieve latencies in the region of 20 ms, which exceeds Wessel and Wright's 10 ms ceiling for musical instruments [10].

Challenges in rich representation. Gesture classification toolkits like `bonk~` have been deployed in many music-making interfaces, including guitars [24], but they were shown to make percussive guitar performers uneasy owing to the chance of misclassification for ambiguous or unexpected inputs [39]. Standard classifiers also do not represent subtle variability within gestural categories, leading to a small gestural *bottleneck* [18]. Related studies in Human-Computer Interaction (HCI) also promote the design of DMIs sensitive to the *micro-scale* of musical actions, the scale of differences across gestures of the same category [40]; authors have suggested that rich and controllable behaviour could be as important as high classification accuracy for creative applications [41]. Dimensionality reduction of input representations through VAEs, as performed for example by RAVE, could help investigate these rich dimensions.

3. METHODOLOGY

3.1 From taxonomy to datasets

This work builds upon our two prior studies on the investigation of the technique of percussive fingerstyle [13] and the design of a prototype augmented guitar to optimally capture those techniques [39]. Those observations firstly

[1] `https://neutone.space/plugin/`

[2] `https://github.com/acids-ircam/nn_tilde`

	Input Features	CNN Type	Bottleneck
TablaCNN	80-band Mel	2-layer 2D. Kernel size: 1x7, 1x3	128 dimensions, reduced through PCA
PercCNN	512-bin FFT decimated to 64 bins	3-layer 1D. Kernel size: 6, 5, 5	2 dimensions
PercVAE	512-bin FFT decimated to 64 bins	3-layer 1D. Kernel size: 6, 5, 5	2 dimensions (μ + σ)

Table 2: Differences across network architectures used.

	Hand Part	Location	In networks
2-class	Kick - K (heel), Non-Kick - NK (all others)	None	*TablaCNN, PercCNN, PercVAE*
4-class	Heel - H, Thumb - T, Fingers - F, Nails - N	None	*TablaCNN, PercCNN, PercVAE*
4-class + 5-class (Hierarchical)	Heel - H, Thumb - T, Fingers - F, Nails - N	Soundhole, Upper Bout, Lower Bout, Upper Side, Lower Side	*TablaCNN, PercCNN*

Table 3: Output layers mapped to guitar body percussion taxonomy.

led to the creation of a *taxonomy of guitar body percussion*, inspired by the work by Goddard on the taxonomy of bass playing techniques [42]:

$$\begin{cases}\text{hit}\\\text{scrape}\end{cases}\ \text{the guitar with}\ \begin{cases}\text{heel}\\\text{thumb}\\\text{fingers}\\\text{nails}\end{cases}\ \text{at the}\ \begin{cases}\text{soundhole}\\\text{upper bout}\\\text{lower bout}\\\text{upper side}\\\text{lower side}\end{cases}$$

This was used to create the labelled dataset *GPercRep* by producing 50 examples (one hit per second) of each combination of taxonomy attributes, excluding those that are ergonomically impossible, e.g. reaching the lower sides of the body with the heel of the hand. This leads to an imbalanced but ecologically valid dataset [43]. Each combination was repeated at four dynamics levels (*p*, *mp*, *mf*, *f*). All recordings were made by the first author on the guitar built for [39], which has a six-channel output made out of one magnetic pickup and five piezo sensors on each of the locations (soundhole, etc..., see taxonomy above). The guitar had 12-53 gauge strings in standard tuning, muted with the left hand, and the hits were played with the bare right hand. After excluding scrapes from the analysis, as they require a time-based gesture follower, the dataset has 3,157 examples extracted from 52 minutes and 37 seconds of audio at 44.1 kHz. The dataset is currently not public.

3.2 Network architectures

The baseline model for our experiments is an adaptation of the tabla transcription model proposed in MA *et al.* [30]. This network processes three stacked spectrograms with different time/frequency resolutions on a window of 150 ms. Each frame of the spectrogram has an 80-bin Mel representation of a window.

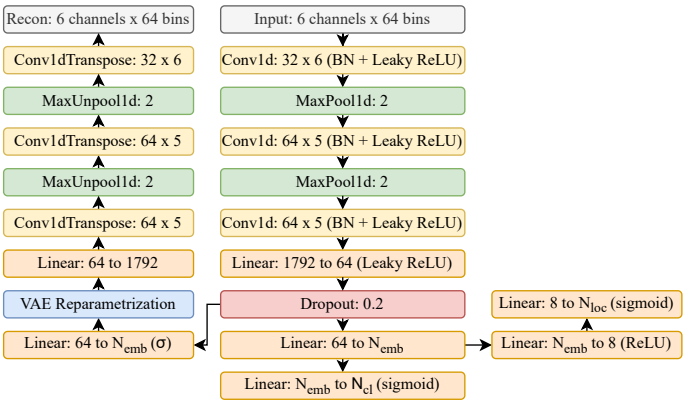

Figure 1: Architecture of *PercCNN*. The extra layers for location classification are on the right-hand side, the decoder of *PercVAE* on the left. $N_{emb} = 2$, $N_{loc} = 5$, $N_{cl} = 2$ or 4.

To adapt this network to real-time requirements we constrained the input window to be 512 samples, or 11.6 ms. Our adaptation (*TablaCNN*) receives one window of six single Mel-frequency spectra, one for each pickup of the prototype. A further modification (*PercCNN*) processes down-sampled FFT features through three one-dimensional convolutional layers and a bottleneck layer of two dimensions before the output (Figure 1). To perform dimensionality reduction jointly with classification, we implemented reparametrisation from the bottleneck layer and a decoder mirroring the encoding CNN (*PercVAE*).

3.3 Output classes

The labels according to the guitar body taxonomy were simplified to: (i) a 2-class scenario with "kicks" (heel hits, in reference to kick drum sounds that heel hits are supposed to imitate) and "non-kicks"; a 4-class output implementing all hand parts; (ii) 4-class hand part plus another 5-class output trained on hit location on the body (hierarchical output). The hierarchical output was not implemented on the VAE. This gave us a total of eight network configurations. Tables 2 and 3 illustrate differences between network architectures, and the mappings between the taxonomy in Section 3.1 and the output layers.

Loss functions used were Binary Cross-Entropy for 2-way classification, Cross-Entropy for 4-way classification, and a sum of two equally weighted Cross-Entropies for hierarchical classification (hand part and location). The VAE used the following loss function, where $\gamma = 0.001$ and $\beta = 3$ after hyperparameter search, and BCE replaced by Cross-Entropy in the four-class model:

$L_{VAE} = BCE + \gamma(MSE_{Recon} + \beta KLD)$

3.4 Training, data augmentation, cross-validation

All networks were trained on the *GPercRep* dataset with hold-out cross-validation: a stratified 20% of the shuffled dataset was reserved for testing, whereas the remainder of

	GPercRep					*GPercHeel*	*GPercPat*		
	K	**NK**	**F**	**N**	**W/Avg**	**Recall**	**K**	**NK**	**W/Avg**
TablaCNN - 2-class	97.46	99.42			99.05	85.69	44.44	85.07	74.56
PercCNN - 2-class	**98.33**	**99.61**			**99.37**	**91.68**	0.00	85.14	63.10
PercVAE- 2-class	97.87	99.51			99.20	85.02	0.00	85.14	63.10
	H	**T**	**F**	**N**	**W/Avg**				
TablaCNN - 4-class	**97.48**	**91.06**	**89.81**	**94.46**	**92.92**	91.35	35.71	87.32	73.97
PercCNN - 4-class	94.61	78.95	80.86	93.26	86.92	69.05	**74.29**	**93.33**	**88.40**
PercVAE - 4-class	97.44	90.30	87.44	93.16	91.63	81.86	0.00	85.14	63.10
TablaCNN - Hierarchical	96.61	92.24	89.59	93.99	92.77	89.18	23.08	86.11	69.80
PercCNN - Hierarchical	95.73	82.05	87.60	94.18	90.12	69.55	0.00	85.14	63.10

Table 4: F-Measure (as percent) for each network on the three test datasets (hold-out of *GPercRep*, *GPercHeel* and *GPercPat*). **H** = heel, **K** = kick, **T** = thumb, **NK** = non-kick, **F** = fingers, **N** = nails, **W/Avg** = weighted average.

the examples was used for training (80%) and validation (20%). All networks were trained with an Adam optimiser for 100 epochs with a batch size of 128, saving the model with the highest accuracy on the test set. We trained with the following data augmentation functions: high-pass at 80 Hz, high-pass at 160 Hz, $tanh()$ waveshaping distortion with gain of 5, phase inversion and random changes in gain between the six channels of each example. Those functions are meant to represent the different input impedances and different gains of other audio preamplifiers.

Further to the *GPercRep* dataset, generalisation was checked by performing cross-dataset evaluation [44]. We recorded a snippet of real-world guitar percussion patterns (*GPercPat*); musically coherent patterns, still with no tonal sounds, were played, rather than hits repeated every second. Acquisition was done with a different audio interface, which led to a different combination of gains and frequency responses across channels in the input audio. The dataset was annotated only with a “kick" and “non-kick” label, leading to 85 hits in one minute of audio.

Testing on the *GPercPat* dataset highlighted a bias in our networks against heel hits or “kicks”. An explanation was thought to be the lack of balance across classes in the dataset, however the issue was not mitigated by balancing the dataset, ensuring the same number of examples for each category. To gather further information about this phenomenon, we created a third dataset consisting exclusively of 601 heel hits, acquired and labelled with the same taxonomy as *GPercRep*: this will be called *GPercHeel*.

3.5 Evaluation metrics

The classification performance of the networks was evaluated with Precision, Recall and F-measure for each category, as a 2-class or 4-class problem (see Table 3).

We also wanted to quantitatively investigate the quality of the network’s embeddings. Thus, we made subsets of the data in *GPercRep* according to each label the network was trained on (kick VS non-kick or hand part), and for each category, we drew distributions for each of the other parts in *GPercRep*’s taxonomy: for example, we divided non-kicks according to their location or their dynamics. Then we calculated the KL-divergences between the probability distributions of each sub-category. The hypothesis behind this method is that, if the embeddings do not carry any meaningful information beyond the classes that the network was trained on, the distributions will overlap and their KL-divergences will be small and noisy. If, on the other hand, different hit properties lead to different positions in the embeddings, KL-divergences will be different across sub-categories and the sub-categories will be arranged following a certain order of similarity.

Reconstruction metrics for the VAE were not evaluated beyond their inclusion in the loss function. Future work could focus on the correlation between better reconstruction and better separation of each feature.

4. EVALUATION

4.1 Classification

Table 4 contains the F-Measure for the predictions of each network, with the three test datasets. In the case of *GPercHeel*, only the Recall is reported; the Precision is always 1, as all hits are heel hits and there cannot be false positives (non-heel hits classified as heel hits).

2-Class discrimination. All networks are able to precisely discriminate between kicks and non-kicks with an F-measure above 99%. The *GPercHeel* dataset shows much reduced but still effective classification, especially with *PercCNN*. However, the test on *GPercPat* exposes a generalisation problem: despite performing data augmentation during training, all networks show a bias toward non-kicks. *PercCNN* and *PercVAE* return only non-kicks in the dataset, despite the two classes being visually separable when data points are extracted and plotted from their embeddings (not pictured). This result may suggest that the networks still overfit to the extent that they are very sensitive to the way that the data is acquired.

4-class discrimination. Uniformly across the tests, the networks yield an F-measure around 90% for *GPercRep*. The introduction of the classification by location (in the two *Hierarchical* networks) does not affect the score of the hand-part classifier. *GPercHeel* yields a similar Recall score, although higher in the case of *TablaCNN* networks. Interestingly, the weakest model in *GPercRep*, the 4-class

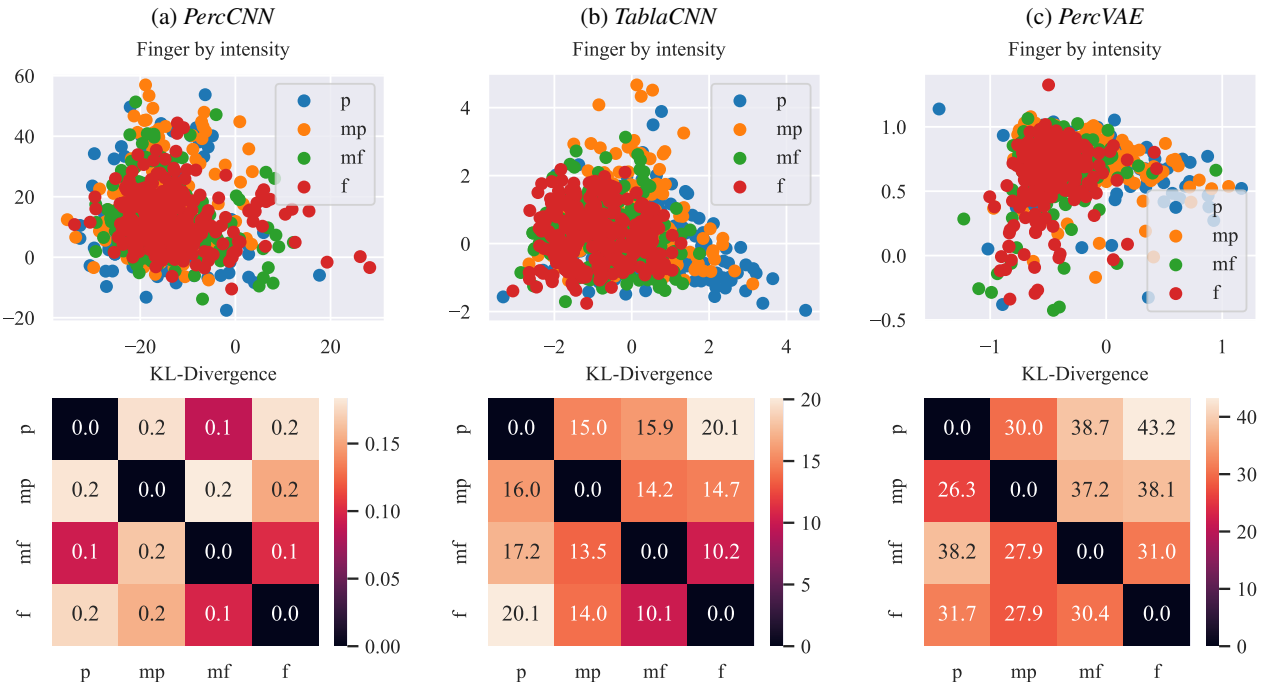

Figure 2: Embeddings from *GPercRep*: example with finger hits labelled by dynamics, with matrix of KL divergence across the distributions of each dynamic level.

PercCNN, ends up being the best model in *GPercPat*, although an F-measure of 74% for kick hits may still not be satisfactory in musical performance. *PercVAE* shows better performance than *PercCNN*, although it still fails to generalise to *GPercPat* and defaults to flagging all events as non-kicks.

The F-Measures in our results are higher on average than the ones found in MA *et al.*'s work on tabla hit transcription [30]. At the same time, our results fall below the 95% accuracy achieved by Stefani [45] with an 8-class discriminator on guitar techniques, and the 97% by Jathal [17] on the three-way discriminator for tabletop drumming. These results, however, are not directly comparable as the figures refer to different datasets.

4.2 Computation times and latency

	Avg	Std Dev
PercCNN	0.496	0.332
TablaCNN	0.422	0.496
PercCNN in Max	12.675	1.132
System (**PercCNN**)	22.310	0.670
System (no NN)	9.922	0.020

Table 5: Computation times in μs of both networks measured through TorchScript in a C++ wrapper, then end-to-end within Max and with an external analogue excitation.

Our models all require a fixed 11.6 ms input buffer to populate the input window after an event is detected, for example through a time-based attack detector [46]. TorchScript was used to wrap the two-class (*PercCNN* and *TablaCNN*) into a C++ test routine and a Max/MSP external for a synthetic soak test and real-world latency tests on a laptop with an Intel i7-8665U CPU running Windows (Table 5).

PercCNN and *TablaCNN* have comparable latencies in the synthetic test. They both execute in less than half a millisecond on average when called 10,000 times. The real-world latency measured manually within Max/MSP (over 30 examples) reports a value that is consistent with the 11.6 ms window plus the synthetic timing reported above, with the attack detection not introducing much further latency or jitter. The low-power laptop used requires an audio buffer size of 256 samples to comfortably run **PercCNN** in real time alongside a suitable synthesis engine: the total system latency jumps to 22 ms when probing with a Bela [3] board attached to the laptop's sound card (averaged over 500 examples). Input and output buffers can be greatly reduced on ad-hoc hardware or software.

4.3 Embeddings

As introduced in Section 3.5, the distribution of subclasses of the taxonomy within each class was explored in the embeddings of each network. In addition to a visual and qualitative inspection of the distribution through scatter plots, KL-Divergence is used here as a similarity metric to measure the distance between distributions. In the following analysis, we will take finger hits divided according to dynamics as an example, but our observations are valid for all other hand parts, and versus hit location (e.g. heel hits

[3] https://bela.io

divided by body location).

Classifier embedding. When *PercCNN* is only trained as a classifier, the four dynamic points overlap in the embeddings (Figure 2a). KL-Divergences range between $2 \cdot 10^{-5}$ and 0.2, so this range of numbers will be used as a baseline for the interpretation of further results. *PercCNNHierarch*, trained to discriminate according to hand part and location, shows very precise segmentation of hit locations but no meaningful segmentation by dynamics (not pictured [4]).

Embedding with PCA. *TablaCNN*'s embeddings are not a bottleneck within the network itself, but they are calculated through PCA on the 128-dimensional dense layer. Principal Component Analysis is shown to disentangle some of the other features in the dataset, as the dynamics subclasses are distributed along a right-to-left gradient (Figure 2b). The KL-Divergence across those distributions reaches a maximum of 20.1.

Embedding/VAE latent space. *PercVAE* shows a similar but more pronounced subdivision in the 2-dimensional latent space. The right-to-left gradient is visible but the KL-Divergence is much greater at a maximum of 43.2. The KL-Divergence values steadily increase from *p* to *f*, more evidently than in the embeddings extracted via PCA. This was noticeable also when hits were segmented by location (not pictured): for example, Lower Side had a KL-Divergence of 30.8 versus Upper Side, 31.7 vs Lower Bout, 38.7 vs Upper Bout, and 40.7 vs Soundhole.

5. DISCUSSION

The evaluation shows that all our models act as very accurate 2-class classifiers. Even though classification accuracy is not as high as in other situations (with different datasets), the simplicity of our models and the 11.6 ms input buffer makes them faster than those systems, and well suited for implementation on an edge device.

Challenges. The main issue arising from our evaluation is the poor generalisation to our *GPercPat* dataset. Still, we have anecdotal evidence that these networks do not behave like poor classifiers in the real-world context of musical performance with our augmented guitar prototype, the HITar [5] . The 2-class *PercCNN* was coupled with a time-domain hit detector and made to run in real time; its continuous output probability was mapped to a linear interpolation of parameters on the modal synthesis engine MetaSynth by CNRS-AMU PRISM [47]; the signal chain was connected to a different guitar (same make and model) to the one the network was trained with; the network is able to reliably adapt synthesis parameters even when used by players other than the main author. There is scope to expand the training and the evaluation by involving more guitars, more players and different data augmentation techniques. However, the augmented guitar that we built allows us to pursue a further type of *behavioural* evaluation with guitar players. In particular, musicians performing in real time may adapt their gestures until they reliably produce a desired set of outcomes, something not possible with prerecorded data. A study on the performance of guitar players with different network configurations running on the augmented guitar prototype will help investigate the degree to which the musicians can adapt to the expectation of the network; such a study would continue our work in [39].

Support for rich interaction. We observed that *PercVAE* is able to encode differences in hit dynamics and location within the embeddings without being trained to discriminate between them; rather than separating them with decision boundaries like *PercCNNHierarch*, each subcategory overlaps neighbouring subcategories, providing a smooth transition that could map well to continuous quantities such as dynamics or location on a surface. The use of a bottleneck layer is also a more efficient solution than PCA, as performing PCA would require extra matrix computation that was not captured in the timings measured at Section 4.2. The parameters of a synthesis engine such as MetaSynth could be controlled not just by the categorical output of the discriminator, but also by the latent representation of the VAE, either directly or through a transform. A mapping function could be designed between the embeddings and synthesis parameters, or the embedding vectors could be exposed directly to synthesisers as MIDI Polyphonic Expression (MPE) [48] controls.

6. CONCLUSIONS

We presented three adaptations of Automatic Drum Transcription for guitar body percussion classification and embedding learning, to support real-time music performance and the augmentation of an acoustic guitar through Deep Neural Networks. We chose and simplified a model for ADT that was shown to be effective in the detection of tabla strokes; a variant was also proposed which supports high-level continuous feature representation through the use of embeddings jointly trained as a Variational Autoencoder's latent space. All network configurations were trained on a dataset of percussive fingerstyle hits acquired *ad hoc*, and they were tested on a hold-out portion of that dataset plus two other datasets of similar material. The networks performed very well on a simplified 2-class discrimination, and comparably to the state of the art on the full 4-class stroke classification with smaller latency. However, they generalise poorly on a dataset that was recorded with different computer equipment. The embeddings were analysed both qualitatively and quantitatively through KL-Divergence between subclasses in the taxonomy; they show that the network encodes some information beyond the categories with which it was trained. We argue that this information can be used to support richness in musical interaction with digital and augmented instruments based on DNN analysis.

[4] All pictures of embeddings available at `https://github.com/iamtheband/martelloni_et_al_ismir2023`

[5] Performance of the HITar at the Guthman Musical Instrument Competition 2023: `https://www.youtube.com/live/NPtHGYH0JV0?t=1150`

HITar's Linktree: `https://linktr.ee/hit4r`

7. ACKNOWLEDGEMENTS

This work was supported by UK Research and Innovation's CDT in AI & Music [grant number EP/S022694/1] and by PRISM Laboratory (CNRS, Aix-Marseille University).

8. REFERENCES

[1] L. Turchet, C. Fischione, G. Essl, D. Keller, and M. Barthet, "Internet of musical things: Vision and challenges," *IEEE access : practical innovations, open solutions*, vol. 6, pp. 61 994–62 017, 2018.

[2] T. Pelinski, V. Shepardson, S. Symons, F. S. Caspe, A. L. Benito Temprano, J. Armitage, C. Kiefer, R. Fiebrink, T. Magnusson, and A. McPherson, "Embedded AI for NIME: Challenges and Opportunities," in *NIME*, Jun. 2022.

[3] D. Stefani, S. Peroni, and L. Turchet, "A comparison of deep learning inference engines for embedded real-time audio classification," in *Int. Conf. on Digital Audio Effects*, vol. 3, 2022, pp. 256–283.

[4] S. Böck, A. Arzt, F. Krebs, and M. Schedl, "Online real-time onset detection with recurrent neural networks," in *Proceedings of the 15th International Conference on Digital Audio Effects (DAFx-12), York, UK*. sn, 2012, pp. 17–21.

[5] C.-Y. Wang, P.-C. Chang, J.-J. Ding, T.-C. Tai, A. Santoso, Y.-T. Liu, and J.-C. Wang, "Spectral–temporal receptive field-based descriptors and hierarchical cascade deep belief network for guitar playing technique classification," *IEEE Transactions on Cybernetics*, vol. 52, no. 5, pp. 3684–3695, 2020.

[6] D. K. Jain, A. Kumar, L. Cai, S. Singhal, and V. Kumar, "ATT: Attention-based timbre transfer," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–6.

[7] F. Caspe, A. McPherson, and M. Sandler, "DDX7: Differentiable FM Synthesis of Musical Instrument Sounds," in *ISMIR*, 2022.

[8] F. T. Liang, M. Gotham, M. Johnson, and J. Shotton, "Automatic stylistic composition of bach chorales with deep LSTM." in *ISMIR*, 2017, pp. 449–456.

[9] D. Stefani and L. Turchet, "On the challenges of embedded real-time music information retrieval," in *DAFx*, 2022.

[10] D. Wessel and M. Wright, "Problems and Prospects for Intimate Musical Control of Computers," *Computer Music Journal*, vol. 26, no. 3, p. 13, 2002.

[11] A. McPherson, R. Jack, and G. Moro, "Action-sound latency: Are our tools fast enough?" in *NIME*, 2016, pp. 20–25.

[12] D. Stowell and M. D. Plumbley, "Delayed Decision-making in Real-time Beatbox Percussion Classification," *Journal of New Music Research*, vol. 39, no. 3, pp. 203–213, Sep. 2010.

[13] A. Martelloni, A. McPherson, and M. Barthet, "Percussive fingerstyle guitar through the lens of NIME: An interview study," in *NIME*, Jul. 2020, pp. 440–445.

[14] F. R. Moore, "The Dysfunctions of MIDI," *Computer Music Journal*, vol. 12, no. 1, p. 19, 1988.

[15] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," *arXiv:1312.6114 [cs, stat]*, May 2014.

[16] M. M. Wanderley, "Gestural control of music," in *International Workshop Human Supervision and Control in Engineering and Music*, 2001, pp. 632–644.

[17] K. Jathal, "Real-Time Timbre Classification for Tabletop Hand Drumming," *Computer Music Journal*, vol. 41, no. 2, pp. 38–51, Jun. 2017.

[18] R. H. Jack, T. Stockman, and A. McPherson, "Rich gesture, reduced control: The influence of constrained mappings on performance technique," in *Proceedings of the 4th International Conference on Movement Computing - MOCO '17*, 2017, pp. 1–8.

[19] P. A. Tremblay and D. Schwarz, "Surfing the Waves: Live Audio Mosaicing of an Electric Bass Performance as a Corpus Browsing Interface," in *NIME*, 2010.

[20] L. Turchet, A. McPherson, and M. Barthet, "Real-Time Hit Classification in a Smart Cajón," *Frontiers in ICT*, vol. 5, Jul. 2018.

[21] B. Zamborlin, "Studies on customisation-driven digital music instruments," Ph.D. dissertation, Goldsmiths, University of London.

[22] M. S. Puckette, T. Apel, and D. D. Zicarelli, "Real-time audio analysis tools for Pd and MSP," in *ICMC*, 1998, p. 5.

[23] R. Fiebrink and P. R. Cook, "The Wekinator: A system for real-time, interactive machine learning in music," in *ISMIR*, 2010.

[24] O. Lahdeoja, "Augmenting Chordophones with Hybrid Percussive Sound Possibilities," in *NIME*, 2009, p. 4.

[25] D. Stefani and L. Turchet, "Demo of the TimbreID-VST Plugin for Embedded Real-Time Classification of Individual Musical Instruments Timbres," in *FRUCT*, 2020, p. 3.

[26] K. Narang and P. Rao, "Acoustic Features for Determining Goodness of Tabla Strokes," in *ISMIR*, 2017.

[27] C.-W. Wu, C. Dittmar, C. Southall, R. Vogl, G. Widmer, J. Hockman, M. Muller, and A. Lerch, "A Review of Automatic Drum Transcription," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 9, pp. 1457–1483, Sep. 2018.

[28] C. Jacques and A. Roebel, "Automatic drum transcription with convolutional neural networks," in *DAFx*, 2018, p. 8.

[29] R. Vogl and P. Knees, "Mirex submission for drum transcription 2018," in *MIREX Extended Abstracts, 19th International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[30] R. MA, A. Bhattacharjee, and P. Rao, "Four-way classification of tabla strokes with models adapted from Automatic Drum Transcription," in *ISMIR*, 2021.

[31] P. Esling, A. Chemla–Romeu-Santos, and A. Bitton, "Generative timbre spaces: Regularizing variational auto-encoders with perceptual metrics," *arXiv:1805.08501 [cs, eess]*, Oct. 2018.

[32] A. Bitton, P. Esling, and A. Chemla-Romeu-Santos, "Modulated Variational auto-Encoders for many-to-many musical timbre transfer," *arXiv:1810.00222 [cs, eess]*, Sep. 2018.

[33] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable Digital Signal Processing," *arXiv:2001.04643 [cs, eess, stat]*, Jan. 2020.

[34] A. Caillon and P. Esling, "RAVE: A variational autoencoder for fast and high-quality neural audio synthesis," *arXiv:2111.05011 [cs, eess]*, Nov. 2021.

[35] J. T. Colonel, C. J. Steinmetz, M. Michelen, and J. D. Reiss, "Direct design of biquad filter cascades with deep learning by sampling random polynomials," *arXiv:2110.03691 [cs, eess]*, Oct. 2021.

[36] F. Eichas, S. Möller, and U. Zölzer, "Block-oriented Gray Box Modeling of Guitar Amplifiers," in *DAFx*, 2017, p. 8.

[37] J. T. Colonel, M. Comunità, and J. Reiss, "Reverse Engineering Memoryless Distortion Effects with Differentiable Waveshapers," in *AES Convention*, Oct. 2022, p. 10.

[38] E.-P. Damskagg, L. Juvela, and V. Valimaki, "Real-Time Modeling of Audio Distortion Circuits with Deep Learning," in *SMC*, 2019.

[39] A. Martelloni, A. McPherson, and M. Barthet, "Guitar augmentation for Percussive Fingerstyle: Combining self-reflexive practice and user-centred design," in *NIME*, Jun. 2021.

[40] J. Armitage, T. Magnusson, and A. McPherson, "Studying subtle and detailed Digital Liutherie: Motivational contexts and technical needs," in *NIME (Accepted)*, 2023.

[41] G. Vigliensoni, P. Perry, and R. Fiebrink, "A Small-Data Mindset for Generative AI Creative Work," in *CHI*, 2022.

[42] C. Goddard, "Virtuosity in computationally creative musical performance for bass guitar," Ph.D. dissertation, Queen Mary University of London, 2021.

[43] D. D. Ramyachitra and P. Manikandan, "Imbalanced Dataset Classification and Solutions: A Review," *International Journal of Computing and Business Research*, vol. 5, no. 4, 2014.

[44] A. Livshin and X. Rodet, "The importance of cross database evaluation in musical instrument sound classification: A critical approach." in *ISMIR*, Jan. 2003.

[45] D. Stefani, S. Peroni, and L. Turchet, "A Comparison of Deep Learning Inference Engines for Embedded Real-time Audio classification," p. 9, 2022.

[46] Luca Turchet, "Hard Real-Time Onset Detection Of Percussive Sounds," in *Proceedings of the 21st International Conference on Digital Audio Effects (DAFx-18)*, 2018, p. 9.

[47] S. Conan, E. Thoret, M. Aramaki, O. Derrien, C. Gondre, R. Kronland-Martinet, and S. Ystad, "Navigating in a space of synthesized interaction-sounds: Rubbing, scratching and rolling sounds," p. 9, 2013.

[48] T. Romo, "MIDI: A Standard for Music in the Ever Changing Digital Age," Capstone Project, California State University, 2018.