Hinrichs *et al.*
*EURASIP Journal on Audio, Speech, and Music Processing* (2022) 2022:28
https://doi.org/10.1186/s13636-022-00257-4

EURASIP Journal on Audio, Speech, and Music Processing

EMPIRICAL RESEARCH

Open Access

# Convolutional neural networks for the classification of guitar effects and extraction of the parameter settings of single and multi-guitar effects from instrument mixes

Reemt Hinrichs*, Kevin Gerkens, Alexander Lange and Jörn Ostermann

**Abstract**

Guitar effects are commonly used in popular music to shape the guitar sound to fit specific genres, or to create more variety within musical compositions. The sound not only is determined by the choice of the guitar effect, but also heavily depends on the parameter settings of the effect. Previous research focused on the classification of guitar effects and extraction of their parameter settings from solo guitar audio recordings. However, more realistic is the classification and extraction from instrument mixes. This work investigates the use of convolution neural networks (CNNs) for the classification and parameter extraction of guitar effects from audio samples containing guitar, bass, keyboard, and drums. The CNN was compared to baseline methods previously proposed, like support vector machines and shallow neural networks together with predesigned features. On two datasets, the CNN achieved classification accuracies 1 — 5 % above the baseline accuracy, achieving up to 97.4 % accuracy. With parameter values between 0.0 and 1.0, mean absolute parameter extraction errors of below 0.016 for the distortion, below 0.052 for the tremolo, and below 0.038 for the slapback delay effect were achieved, matching or surpassing the presumed human expert error of 0.05. The CNN approach was found to generalize to further effects, achieving mean absolute parameter extraction errors below 0.05 for the chorus, phaser, reverb, and overdrive effect. For sequentially applied combinations of distortion, tremolo, and slapback delay, the mean extraction error slightly increased from the performance for the single effects to the range of 0.05 to 0.1. The CNN was found to be moderately robust to noise and pitch changes of the background instrumentation suggesting that the CNN extracted meaningful features.

**Keywords:** Convolutional neural networks, Guitar effects, Parameter extraction, Music information retrieval

## 1 Introduction

Audio effects are a wide-spread tool used in the production and creation of music [1]. They find application in all kinds of music and are applied to virtually all kinds of instruments such as vocals, guitar, keyboard, and so on [2]. For example, in the domain of guitar-centered music, a prominent and well-known effect is the overdrive effect, closely related to the distortion effect. A multitude of other effects exist such as phaser, delay, ring-modulator, and many more [3]. Several professional guitarists use guitar effects to create a unique, distinctive sound, strongly associated with the artist, a prime example being U2's the Edge. For the production of music, and the creative process of writing music, the automatic creation of guitar effect settings that yield a desired sound can be

*Correspondence: hinrichs@tnt.uni-hannover.de

Institut für Informationsverarbeitung, Leibniz University Hannover, Hannover, Germany

of interest. For this purpose, extraction algorithms are required, which map audio recordings or signals to effect classes and associated effect parameter settings.

### 1.1 Background

Early work in this domain focused solely on the classification of guitar effects. Stein et al. pioneered this area of research with their fundamental work [4, 5], using a large set of audio features and a support vector machine to classify eleven guitar effects. They achieved an accuracy of 97.7 % for solo guitar recordings.

Further work regarding the classification of guitar effects was done by Eichas et al. [6] and Schmitt et al. [7], the latter investigating the importance of audio features and comparing the so called bag-of-audio-words approach to the use of functionals. They found both approaches to achieve similar high performance. The issue of deriving or extracting the parameter settings of a guitar effect from audio samples is closely related to sound matching [8], the estimation of synthesizer parameters replicating a certain given sound. Yee-King et al. [9] proposed a special long short-term memory network structure for this purpose and achieved a close sound for 25 % of their test set. Sheng and György [10] investigated the extraction of dynamic range compression parameter settings from audio samples of violins and drums using several audio features and regression models. For the same purpose, the authors investigated deep neural networks [11] and found them to improve performance, predicting multiple dynamic range compression parameters at once from monophonic and polyphonic audio samples.

Research regarding the extraction of guitar effect parameter settings is scarce. So far, only three previous works exist: Jürgens et al. [12] pioneered this task using shallow neural networks combined with specifically designed features for each guitar effect, achieving or surpassing the (presumed) performance of a human expert. Comunità et al. [13] used convolutional neural networks (CNNs) to extract the parameter settings of different implementations of distortion-related guitar effects from monophonic and polyphonic audio samples, achieving below 0.1 root-mean-square error in all cases. In [14], a CNN was used for the classification of guitar effects from instrument mixes as well as the extraction of their parameter settings. The CNN was used for the extraction of single guitar effects limited to distortion, tremolo, and slapback delay. The CNN yielded presumed human expert-level results for all effects considered. Except for [14], none of these research papers considered extraction of guitar effect parameter settings from instrument mixes, i.e., audio recordings or signals, in which several instruments play at once.

### 1.2 Contribution

This manuscript expands our results of [14] in several ways: (i) multi-effects are also considered, i.e., sequences of guitar effects applied one after the other; (ii) results are expanded to four additional guitar effects, namely chorus, reverb, overdrive, and phaser, proving the versatility of the CNN; and (iii) the error analysis is more indepth.

For this purpose, a custom dataset was created consisting of instrument mixes of guitar, bass, keyboard, and drums using virtual instruments. As baseline approach for effect classification, we used the method of Stein et al. [4], and for extraction of the effect parameter settings, we used the method of Jürgens et al. [12] and compared it to the performance of a CNN at different volume levels of the instrument mix. Four different time-frequency representations — the chromagram, the spectrogram, the mel-frequency cepstral coefficients (MFCCs), and gammatone frequency cepstral coefficients (GFCCs) — were assessed with respect to the achieved CNN performance.

Because guitar effects can and are used often in conjunction ("effect chains"), we investigated the extraction of the parameter settings for single guitar effects as well as for up to three guitar effects at once. The focus in the extraction of the guitar effect parameter settings lied on the distortion, tremolo, and slapback delay effect to allow a comparison to the method by Jürgens et al. [12]. Additionally, using the knowledge of the previous experiments, we considered the phaser, chorus, reverb, and overdrive effect as well. Because a common issue regarding artificial neural networks is their robustness [15, 16] and proneness to over-fitting [17], the sensitivity of the CNN towards noise on the input data as well as different pitches of the background instrumentation was assessed as well.

To shorten the phrasing a little, guitar effect parameter setting extraction will be called effect parameter extraction or just parameter extraction.

Section 2 describes the datasets used for classification and parameter extraction, the time-frequency representations used, and the training and evaluation of the CNNs. Section 3 reports the classification, extraction, and robustness results. Section 4 discusses and interprets these results and the manuscript is concluded in Section 5.

## 2 Method and materials

This section first explains in detail the creation of the used datasets. Then, the investigated time-frequency representations as well as the architecture of the convolutional neural networks are discussed. Finally, the robustness and error analysis is presented.

### 2.1 Dataset

Two datasets were created specifically for the investigations of this work, one for guitar effect classification, abbreviated GEC-GIM (for Guitar Effect Classification — Guitar Instrument Mix), and one for guitar effect parameter extraction, abbreviated GEPE-GIM (for Guitar Effect Parameter Extraction — Guitar Instrument Mix). The motivation to create two separate datasets was the large number of samples that had to be created, if the same amount of parameter settings used for GEPE-GIM had been used for GEC-GIM. The very same effect plugins of Stein et al. [4] and Jürgens et al. [12] were used in this work and are listed in Table 1. Effect descriptions and their parameters can be found in [3, 4, 12]. Additionally, to further test the capability of the CNN, the monophonic guitar samples of the IDMT-SMT dataset from Stein et al. [12] were used for classification.

GEC-GIM, the dataset for guitar effect classification from instrument mixes, used exactly one fixed (relative) mixing volume of 0 dB, where the mix was created as described shortly; however, in contrast, in total, eleven guitar effects were used. For each combination of guitar plugin, instrument mix, and guitar effect, 60 random settings of the effect parameters were used. This resulted in 1440 samples for each effect. The dataset consisted of twelve instrument combinations of the guitar and the other instruments, i.e., guitar and keyboard, guitar and bass, guitar and bass and keyboard, and guitar and bass and keyboard and drums. It also covered solo guitar as well as guitar together with individual drum parts, i.e., guitar and snare and guitar and crash cymbal. In total, this dataset contained 15,840 audio samples.

GEPE-GIM, the dataset used to investigate parameter extraction, consisted of instrument mixes of guitar, keyboard, bass, and drums. It consisted of single guitar effect samples, where only one individual guitar effect was applied to the guitar signal, and multi-effect guitar samples, where two or three guitar effects were applied sequentially forming an effect chain.

All instruments were virtual instruments and created using the following plugins: Sonivox Bright Electric Guitar and Ample Guitar LP (guitar), Bitsonic Keyzone Classic (keyboard), Ample Bass P Lite (bass), and Manda Audio MT POWER DrumKit 2 (drums). All plugins are sample-based and thus could be expected to create realistic waveforms/sounds. The IDMT-SMT dataset was not considered for the creation of these datasets, because the guitar samples vary considerably in their initial silence. Due to that, the instrument mixes would sound odd without further processing. This would have complicated the dataset creation (using midi data) and the investigations without a clear advantage. The chosen guitar plugins are also using real recordings of electric guitars and thus are just as realistic as the recordings of the IDMT-SMT dataset.

While guitar, keyboard, and bass played a single note in each sample, starting at the exact same time lasting for 2 s, the drums played the pattern depicted in Fig. 1. The guitar and keyboard played the notes E2 and E3, and bass played the notes E1 and E2. The guitar note was

**Table 1** Overview of the used guitar effect plugins. The variable parameters are those varied to generate the audio samples of the datasets. To generate the slapback delay effect, the feedback delay with its feedback parameter set to zero was used

| Manufacturer | Plugin name | Effect type | Variable parameters |
|---|---|---|---|
| Kjaerhus Audio [18] | Classic Chorus | Chorus | Rate, depth |
| | | Vibrato | Rate, depth |
| | Classic Delay | Feedback delay | Time, feedback, mix |
| | | Slapback delay | Time, mix |
| | Classic Flanger | Flanger | Rate, depth, feedback |
| | Classic Phaser | Phaser | Rate, depth |
| | Classic Reverb | Reverb | Room size, mix |
| SimulAnalog [19] | RednefTwin | Equalizer | Bass, mids, treble |
| | Tube Screamer | Overdrive | Gain, tone |
| PechenegFX [20] | Pecheneg Tremolo | Tremolo | Rate, depth |
| Buzzroom [21] | OctBUZ | Distortion | Gain, tone |

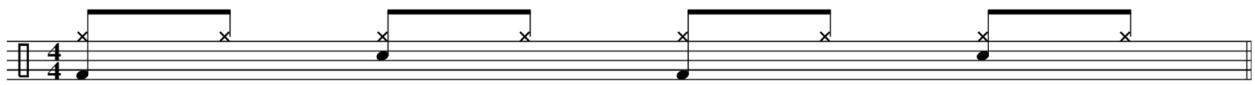

**Fig. 1** Tablature showing the drum pattern used in all samples. All other instruments played a single note for the duration of 2 s, i.e., one bar

independently varied from keyboard and bass; however, keyboard and bass notes were always moved together by an octave to reduce the amount of data.

Keyboard, bass, and drums were mixed at different volume levels with the guitar according to

$$\text{premix}(t) = b(t) + k(t) + d(t) \tag{1}$$

and

$$\text{mix}(t) = g(t) + \alpha \cdot \text{premix}(t), \tag{2}$$

where $b(t)$, $k(t)$, $d(t)$, and $g(t)$ are the bass, keyboard, drum, and guitar signals, respectively. All guitar effects were applied to the signal $g(t)$ before mixing. The parameter $\alpha$ controlled the (relative) mixing volume and was set such that

$$\beta = 20 \cdot \log_{10} \frac{\max_t \alpha |\text{premix}(t)|}{\max_t |g(t)|} \tag{3}$$

equaled the desired mixing volumes. The mixing volumes used were $\beta = -36\,\text{dB}, -24\,\text{dB}, -12\,\text{dB}, -6\,\text{dB}, -3\,\text{dB}, 0\,\text{dB}$ and $\beta = +3\,\text{dB}$. After mixing, the waveforms $\text{mix}(t)$ were peak normalized, yielding the final audio samples. Subjectively, the mix was considered realistic in the sense that, e.g., keyboard and bass were clearly and loudly audible, not overshadowing each other, and the guitar clearly moved towards the background with increasing mixing volume. An example waveform of an audio sample with tremolo on the guitar showing the lowest and highest mixing volume after normalization is depicted in Fig. 2. For each of these guitar effects, their parameters were varied in steps of 0.05, starting at 0.05 and ending at 1, creating a grid of parameter values with 400 tuples of parameter settings. Thus, for each mixing volume, $2^3 \cdot 400 = 3200$ samples per guitar effect were generated, where the multiplication with $2^3$ is due to the two guitar plugins and the two octaves played by guitar, keyboard, and bass.

To create multi-effect samples, distortion, tremolo, and slapback delay were applied in conjunction, one after the other. All possible combinations were considered. The order in which the effects were applied (and bypassed if not present) was not changed, i.e., always distortion first, tremolo second and slapback delay last. In contrast to the generation of the single effect samples, for the generation of multi-effect samples using two effects, 800 random tuples of parameter settings were used. For three effects, 1200 random tuples of parameter settings were used. The reason was to reduce the total amount of data.

Finally, to extend the investigation beyond the guitar effects considered by Jürgens et al. [12], we included four additional guitar effects in the dataset, namely chorus, phaser, overdrive, and reverb. No effect combinations were considered in this case. Single and multi-effects combined, 268,800 samples were created for the investigation of parameter extraction.

All audio waveforms were sampled at 44.1 kHz and had a duration of 2 s, each corresponding to four quarter notes or one bar at 120 beats per minute and four quarters time.

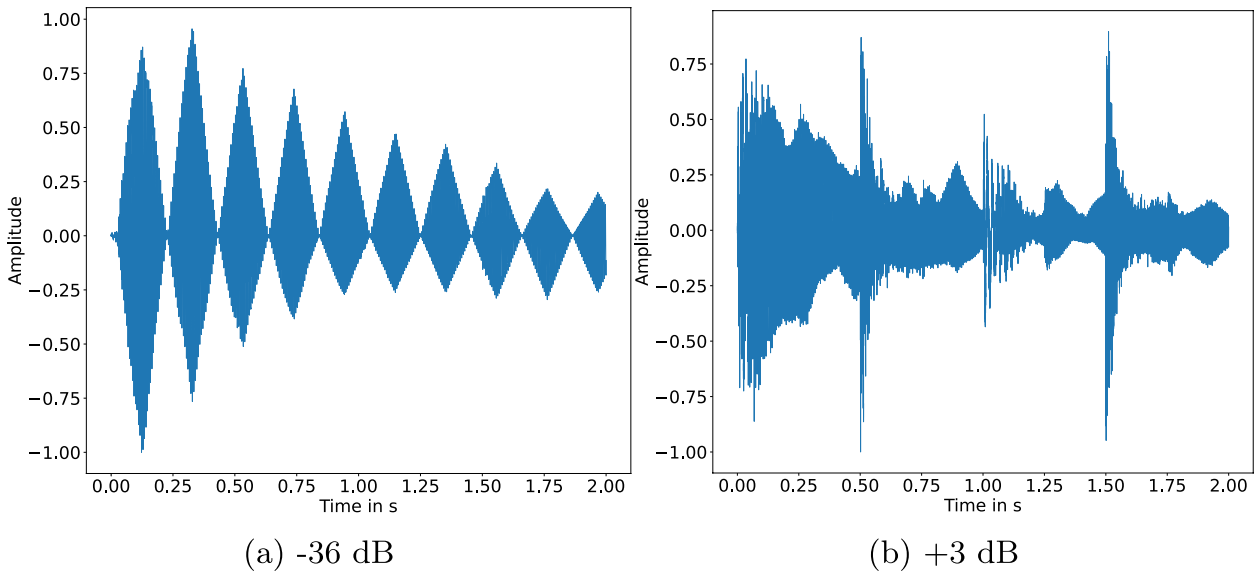


**Fig. 2** Audio sample with tremolo on the guitar at **a** lowest (−36 dB) and **b** highest (+3 dB) mixing volume after peak normalization. The two large peaks in **b** are due to the snare hits. Modulation depth of the tremolo effect was 1.0, and modulation frequency was 0.15

### 2.2 Architecture

Four different time-frequency representations were investigated as input of the CNNs. These were the (magnitude) spectrogram, the chromagram, mel-frequency cepstral coefficients (MFCCs), and gammatone frequency cepstral coefficients (GFCCs). The chromagram is a mapping of an audio waveform to the twelve semitones of the western music that yields the energy in the respective semitones. See [22] for a description of the chromagram and the spectrogram and [23] for a description of the MFCCs and GFCCs. While the spectrogram and MFCCs were chosen due to their widespread use and known good performance, the GFCCs were chosen because some publications [24, 25] find them to be slightly superior to MFCCs in audio classification tasks. Finally, the chromagram was considered due to its widespread usage in music signal processing.

For the MFCCs and GFCCs, 40 coefficients were used, denoted MFCC40 and GFCC40 in the figures. For the computation of the GFCCs, the sampling rate had to be reduced to 16 kHz. An example of the four time-frequency representations applied to the same audio sample is shown in Fig. 3. The dimensions of the images, obtained by applying the time-frequency representations to the audio samples, were 256 × 173 (spectrogram), 40 × 173 (MFCCs), 12 × 173 (chromagram), and 40 × 193 (GFCCs). The time-frequency representations were normalized before being passed to the CNN using sklearn's Standard Scalar class.

Two slightly different CNNs were used, one for effect classification and one for the parameter extraction. The CNN structures are given in Table 2 and were based on [13]. The total number of weights of the CNN used for classification ranged from 192,587 (chromagram) to 10,436,683 (spectrogram) with 1,368,139 for MFCCs and 1,597,515 for GFCCs. The total number of weights of the CNN used for parameter extraction ranged from 37,146 (chromagram) to 1,957,914 (spectrogram) with 257,562 for MFCCs and 300,570 for GFCCs. In the case of effect classification, the output dimension was eleven, matching the number of effects. In the case of parameter extraction, the output dimension for single effect parameter extraction was two, matching the number of parameters per effect. Accordingly, the output dimensions were four and six for multi-effect parameter extraction.

The CNNs were trained for 70 epochs with a learning rate of 0.001 using the adam solver for single effect parameter extraction. For classification and multi-effect parameter extraction, 100 epochs were used. The loss functions for the classification and extraction of the parameters were the cross-entropy loss and mean squared error, respectively. A 80%/20% training/test split of the datasets was used where the test set was only used for evaluating the CNN performance and did not influence the training process in anyway. Five-fold cross-validation was performed and each training used a new training/test split. The batch size was set to 64 for classification and 128 for parameter extraction.

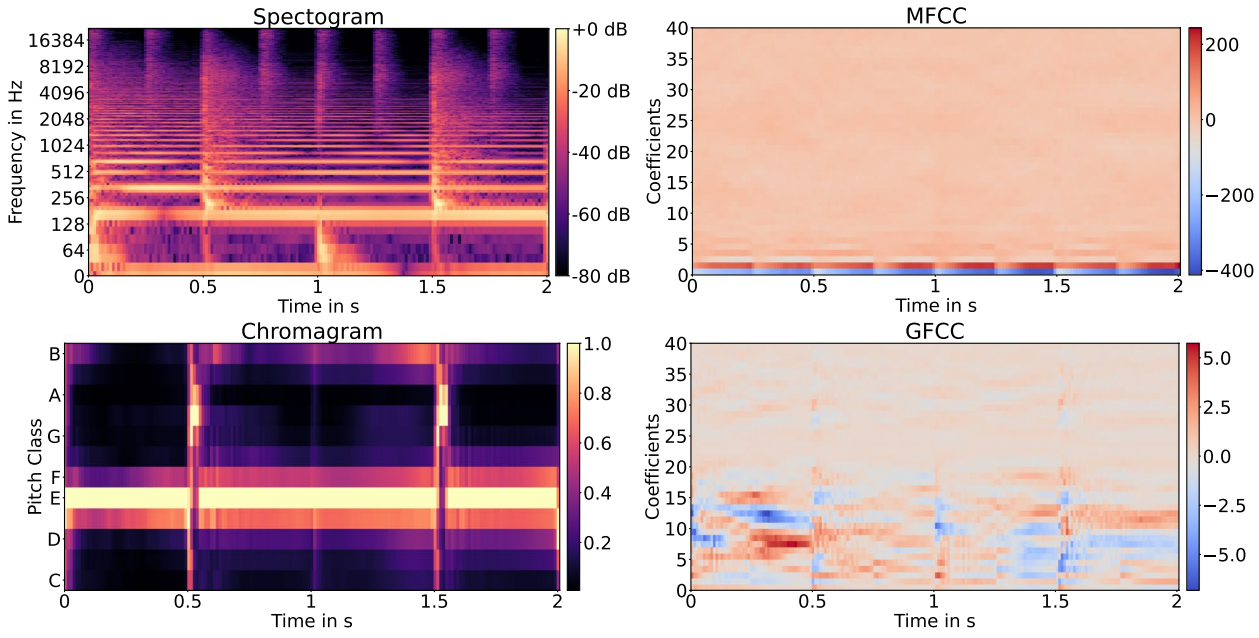


**Fig. 3** Example of all four investigated time-frequency representations of one sample of the GEPE-GIM dataset with distortion on the guitar and gain set to 0.7 and tone set to 0.85 at a mixing volume of +3 dB

**Table 2** Structure of the convolutional neural networks (CNNs) used for (a) effect classification and (b) parameter extraction. Each guitar effect investigated had two parameters resulting in an output dimension of two for the CNN for extraction of the guitar effect settings

| Layer | Kernel | Filter | Activation | Dropout |
|---|---|---|---|---|
| (a) CNN structure for effect classification | | | | |
| Convolutional | 3 × 3 | 32 | ReLU | - |
| Batch norm. | - | - | - | - |
| Max pooling | 2 × 2 | - | - | - |
| Convolutional | 3 × 3 | 64 | ReLU | 0.3 |
| Batch norm. | - | - | - | - |
| Max pooling | 2 × 2 | - | - | - |
| Flatten | - | - | - | - |
| Dense | - | 64 | ReLU | 0.3 |
| Batch norm. | - | - | - | - |
| Dense | - | 64 | ReLU | 0.3 |
| Batch norm. | - | - | - | - |
| Dense (output) | - | 11 | Softmax | - |
| (b) CNN structure for effect parameter extraction | | | | |
| Convolutional | 3 × 3 | 6 | ReLU | - |
| Batch normalization | - | - | - | - |
| Max pooling | 2 × 2 | - | - | - |
| Convolutional | 3 × 3 | 12 | ReLU | 0,2 |
| Batch normalization | - | - | - | - |
| Max pooling | 2 × 2 | - | - | - |
| Flatten | - | - | - | - |
| Dense | - | 64 | ReLU | 0,2 |
| Batch normalization | - | - | - | - |
| Dense | - | 64 | ReLU | 0,2 |
| Batch normalization | - | - | - | - |
| Dense (output) | - | 2 | Sigmoid | - |

### 2.3 Experimental setup

#### *2.3.1 Baseline*

For effect classification, we used the support vector machine (SVM) classifier as described in [4]. It was later also tested in [12]. In our case, the onset detection was removed as the onset was always the same due to using virtual instruments. Using the features and functionals proposed in [4], a total of 649 functionals were used. For effect parameter extraction, the method proposed by Jürgens et al. [12] served as the baseline. Furthermore, the presumed human expert setup error of 0.05 served as a baseline for parameter extraction. This value, first used in [12], was derived informally, i.e., without performing rigorous listening tests. It stems from the experience of the authors, which recognized that usually guitar effect (plugin) parameters are set in 0.05 steps or larger and smaller step sizes usually are indistinguishable. The impact of background instrumentation on this reference is unknown and difficult to judge; however, it appears reasonable that, given a sufficiently long audio sample, human hearing should be rather robust towards mixing volume.

#### *2.3.2 Robustness analysis*

As artificial neural networks are prone to overfitting or to fit to unexpected patterns in the data [16, 26], the robustness of the CNN for parameter extraction was analyzed. For this purpose, white Gaussian noise with zero mean and a standard deviation $\sigma_s$, set according to

$$\sigma_s = \alpha \cdot \max\{|C_s(t,f)|\}, \tag{4}$$

with $\alpha \in \{0, 0.001, 0.01, 0.05\}$ was added to the time-frequency representations after normalization. $C_s(t,f)$ denotes the respective time-frequency representations. We then assessed the impact of this noise on parameter extraction of the CNN. The maximum in Eq. 4 was taken across the entirety of the dataset. An $\alpha$ value of zero, included as reference, corresponded to the original, noise-free samples. Additionally, the CNN was tested with novel tones of keyboard and bass, which now were moved in semitone steps from $E2$ (keyboard) and $E1$ (bass) to the next octave, and the impact on the parameter extraction error was assessed. An example of the impact of the noise on the spectrogram for $\alpha = 0.001$ is depicted in Fig. 4.

#### *2.3.3 Error analysis*

We also investigated the dependence of the parameter extraction errors of the CNNs on the true parameter settings. Large errors occurring for parameter settings, where one parameter has no to little effect on the audio (e.g., tone parameter when the gain parameter is near zero for the distortion effect), can be attributed to subjective equivalence and not failures of the CNN. Then, extraction errors are of little subjective relevance and the extracted parameter settings are still useful. We considered only the maximum extraction error across the 5-fold cross-validation, and not, e.g., the mean, to reduce the amount of figures.

## 3 Results

The results are presented in several steps: Firstly, the effect classification is presented. Secondly, the effect parameter extraction for individual effects is presented. Thirdly, the effect parameter extraction for multi-effects is presented. Then, results for the chorus, phaser, reverb, and overdrive effect are given. Finally, results regarding the robustness of the CNN and the error analysis are presented.

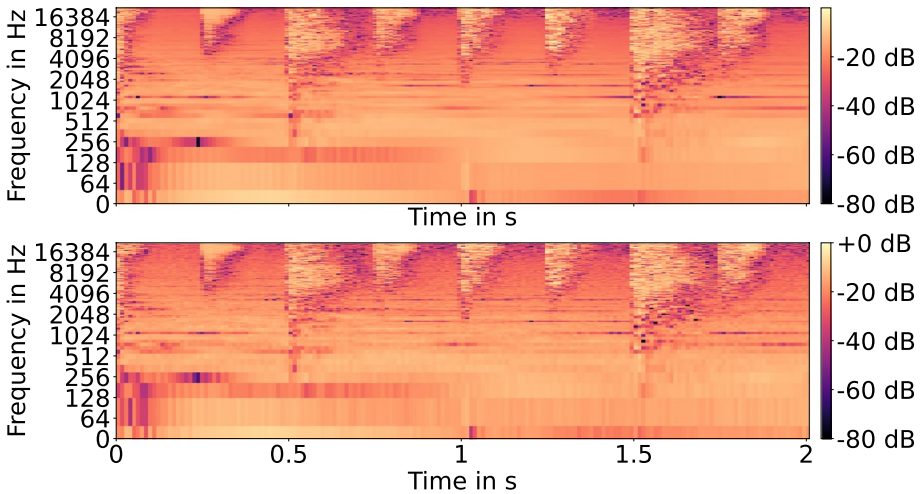


**Fig. 4** Spectrogram of an audio sample at 0 dB mixing volume using distortion with gain set to 0.15 and tone set to 0.8 after normalization of the dataset as explained in Section 2.2 without (top) additional noise, i.e., $\alpha = 0$, and with (bottom) additional noise with $\alpha = 0.001$and $\alpha$ as explained in Section 2.3.2. The noise is noticeable mostly at low magnitudes

### 3.1 Effect classification

Figure 5 depicts the confusion matrices of the SVM and CNN classifiers on the IDMT-SMT and GEC-GIM datasets. Accuracies are summarized in Table 3. For the CNN, the depicted confusion matrices were achieved using GFCCs (IDMT-SMT) and the spectrogram (GEC-GIM). The accuracies for the other time-frequency representations were similar, except for the GFCCs on the GEC-GIM dataset where the CNN failed to converge. While no obvious reason was found, inspection of the loss curves suggested an insufficient learning rate. However, because the GFCCs otherwise performed very well, this issue was not investigated further. On the GEC-GIM, both classifiers tended to confuse slapback and feedback delay which greatly impacted the overall accuracies. However, it was later recognized that, at least for settings below 0.3 of the feedback parameter, a range which due to chance about 30 % of the parameters settings belonged to, the feedback and slapback delay were very difficult to distinguish as confirmed by selected listening tests. This issue did not arise on the IDMT-SMT, which uses rather distinct settings for slapback and feedback delay. On both datasets, the CNN outperformed the SVM classifier with an accuracy of up to 90.02% for the CNN on the GEC-GIM and 85.01% for the SVM. On the IDMT-SMT, the CNN achieved up to 97.38% accuracy compared to 96.16% accuracy for the SVM classifier. As the slapback delay is identical to the feedback delay when the feedback parameter is set to zero, confusing these two effects is not necessarily a problem for subsequent parameter extraction. Due to this, the classification accuracy, when these two are considered as the same effect, was specified as well, increasing the accuracies by about 4–5 % in all cases.

### 3.2 Effect parameter extraction of single effects

Boxplots of the absolute extraction error across volume for the CNN as well as the method by Jürgens et al. [12] are shown for the distortion, tremolo, and slapback delay effect in Fig. 6. The mean absolute extraction error across volume is summarized in Table 4. No single time-frequency representation was optimal irrespective of the considered guitar effect. The CNN using MFCCs achieved the minimum mean absolute error of below 0.017 for the distortion effect and either parameter. The GFCCs worked best for the slapback delay, yielding a mean absolute error below 0.04 for either parameter. The outliers occurred mostly for parameter settings that were very difficult to distinguish as will be discussed shortly.

The mean absolute extraction error across volume for the slapback delay for all time-frequency representations is shown in Fig. 7. The other effects qualitatively showed similar dependency on the volume. Usually, a two- or threefold increase from the lowest to the highest mixing volume was observed, albeit even at the highest mixing volume of +3 dB, the error was, on average, equalling or surpassing human expert level. Figure 8 depicts a comparison of the CNN using MFCCs and the method by Jürgens et al., both evaluated across volume. The method by Jürgens et al. was less affected by the volume, increasing by at most about

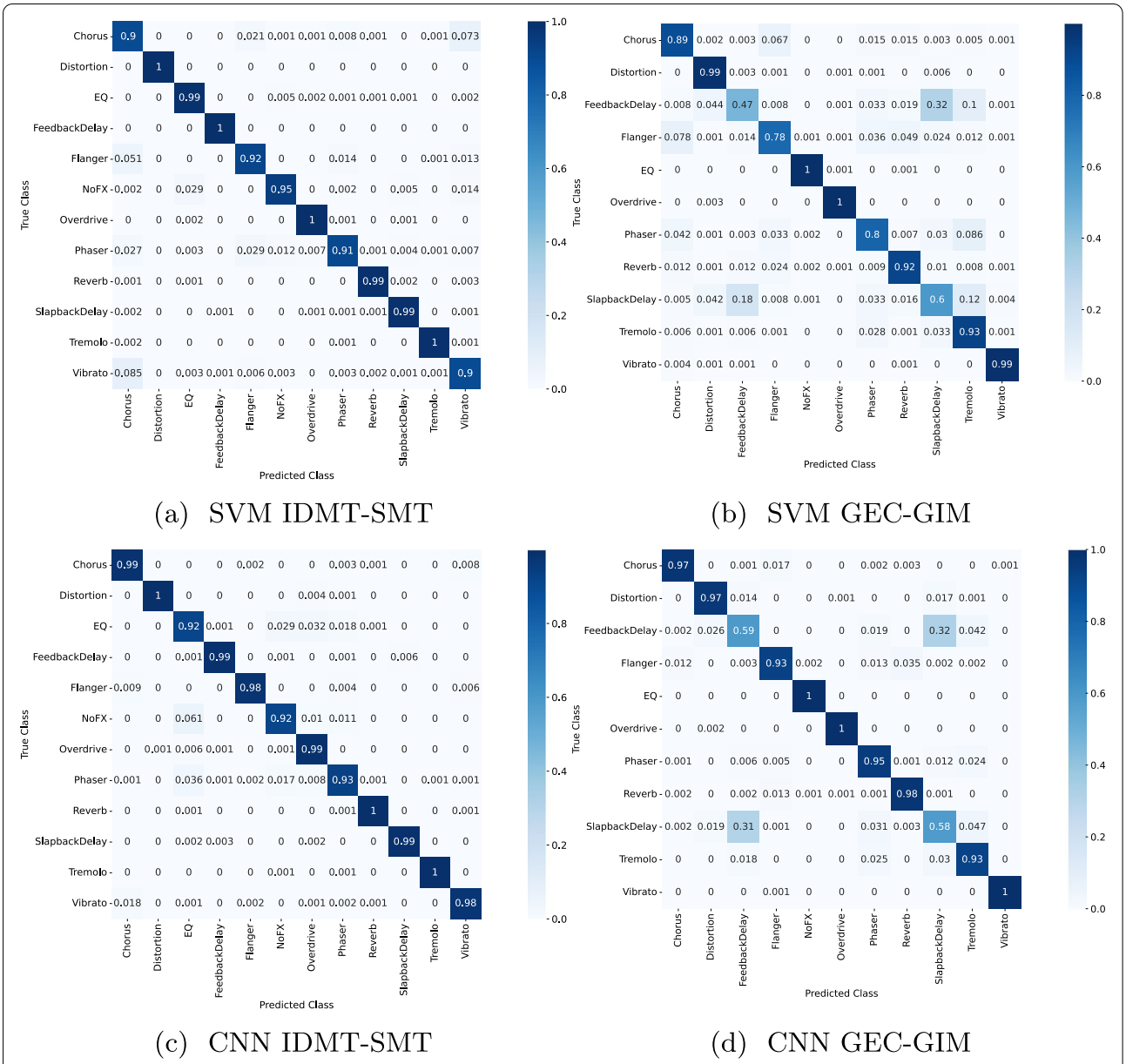


**Fig. 5** Confusion matrices of the support vector machine classifier (SVM) on the **a** IDMT-SMT and **b** GEC-GIM dataset as well as the convolutional neural network (CNN) classifier on the **c** IDMT-SMT and **d** GEC-GIM dataset. On the GEC-GIM, both classifiers tend to confuse slapback and feedback delay which was due to random settings of the feedback parameter. For feedback settings below at least 0.3, slapback and feedback delay indeed were very difficult to distinguish which was confirmed through selected listening tests

50% in mean absolute error with increasing volume. Still, its performance was considerably worse than the CNN at the highest volume.

### 3.3 Effect parameter extraction of multi-effects

The 95 % confidence intervals for the mean absolute error of the parameter extraction are given for the combinations of distortion and tremolo in Table 5, for the distortion and slapback delay in Table 6, and for the combination of tremolo and slapback delay in Table 7. For the combination of distortion, tremolo, and slapback delay, results are reported in Table 8. Boxplots comparing the absolute extraction error achieved for individual effects and effect combinations are depicted in Fig. 9. It was observed that, except for the distortion effect, the mean absolute error of the parameter extraction

**Table 3** 95 % confidence intervals for the classification accuracy of the convolutional neural network (CNN) using the listed time-frequency representations as well as the accuracy of the baseline support vector machine (SVM) using the method by Stein et al. [4]. As subsequent parameter extraction is not necessarily compromised by a confusion of slapback delay (SD) and feedback delay (FD), the respective accuracies, when SD and FD are treated as the same effect, are given as well

| Method | GEC-GIM | GEC-GIM (SD = FD) | IDMT-SMT |
|---|---|---|---|
| SVM | 85.0 % ± 0.44 % | 89.4 % ± 0.32 % | 96.1 % ± 0.3 % |
| CNN + spectrogram | **90.0** % ± 0.57 % | **95.8** % ± 0.52 % | **97.4** % ± 0.7 % |
| CNN + MFCCs | 87.7 % ± 0.52 % | 93.4 % ± 0.32 % | 96.5 % ± 0.13 % |
| CNN + GFCCs | 24.7 % ± 27.3 % | 28.7 % ± 30.47 % | **97.4** % ± 0.3 % |
| CNN + chromagram | 87.0 % ± 0.64 % | 92.9 % ± 0.12 % | 86.2 % ± 0.2 % |

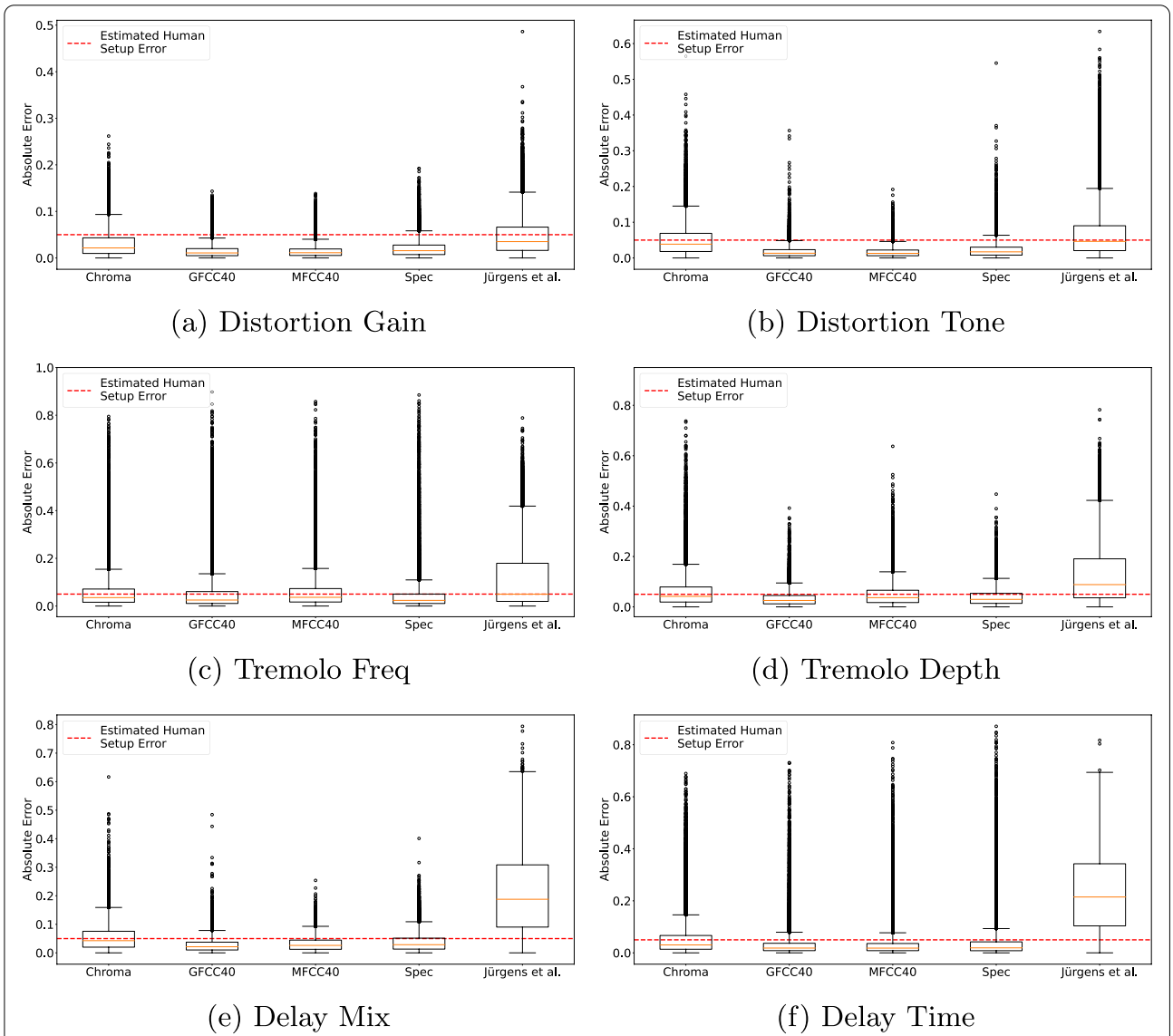


**Fig. 6** Boxplots of the absolute parameter extraction error for the **a** distortion gain, **b** distortion tone, **c** tremolo frequency, **d** tremolo depth, **e** delay mix, and **f** delay time parameter for all investigated methods across all volumes

**Table 4** 95 % confidence interval of the mean absolute error of the CNN across the fivefold cross-validation of the parameter extraction and all volumes for the individual effects. Results are given for Jürgens et al. [12] as well. The lowest errors are highlighted using bold font

| | **Distortion** | | **Tremolo** | | **Slapback delay** | |
|---|---|---|---|---|---|---|
| | **Gain** [$10^{-2}$] | **Tone** [$10^{-2}$] | **Depth** [$10^{-2}$] | **Freq.** [$10^{-2}$] | **Time** [$10^{-2}$] | **Mix** [$10^{-2}$] |
| CNN + chromagram | $3.2 \pm 0.15$ | $5.1 \pm 0.48$ | $6.3 \pm 0.17$ | $6.1 \pm 0.38$ | $6.2 \pm 0.25$ | $5.4 \pm 0.3$ |
| CNN + GFCC | $1.6 \pm 0.25$ | $1.7 \pm 0.15$ | $\mathbf{3.4} \pm 0.2$ | $6.3 \pm 0.33$ | $\mathbf{3.8} \pm 0.07$ | $\mathbf{2.7} \pm 0.14$ |
| CNN + MFCC | $\mathbf{1.4} \pm 0.09$ | $\mathbf{1.6} \pm 0.09$ | $4.9 \pm 0.45$ | $6.5 \pm 0.15$ | $3.9 \pm 0.1$ | $3.1 \pm 0.22$ |
| CNN + spectrogram | $2.1 \pm 0.63$ | $2.3 \pm 0.18$ | $3.9 \pm 0.25$ | $\mathbf{5.2} \pm 0.59$ | $4.9 \pm 0.09$ | $3.7 \pm 0.47$ |
| Jürgens et al. | $4.8 \pm 0.82$ | $7.1 \pm 2.4$ | $12.9 \pm 3.4$ | $11.8 \pm 2.08$ | $22.8 \pm 7.6$ | $20.07 \pm 6.1$ |

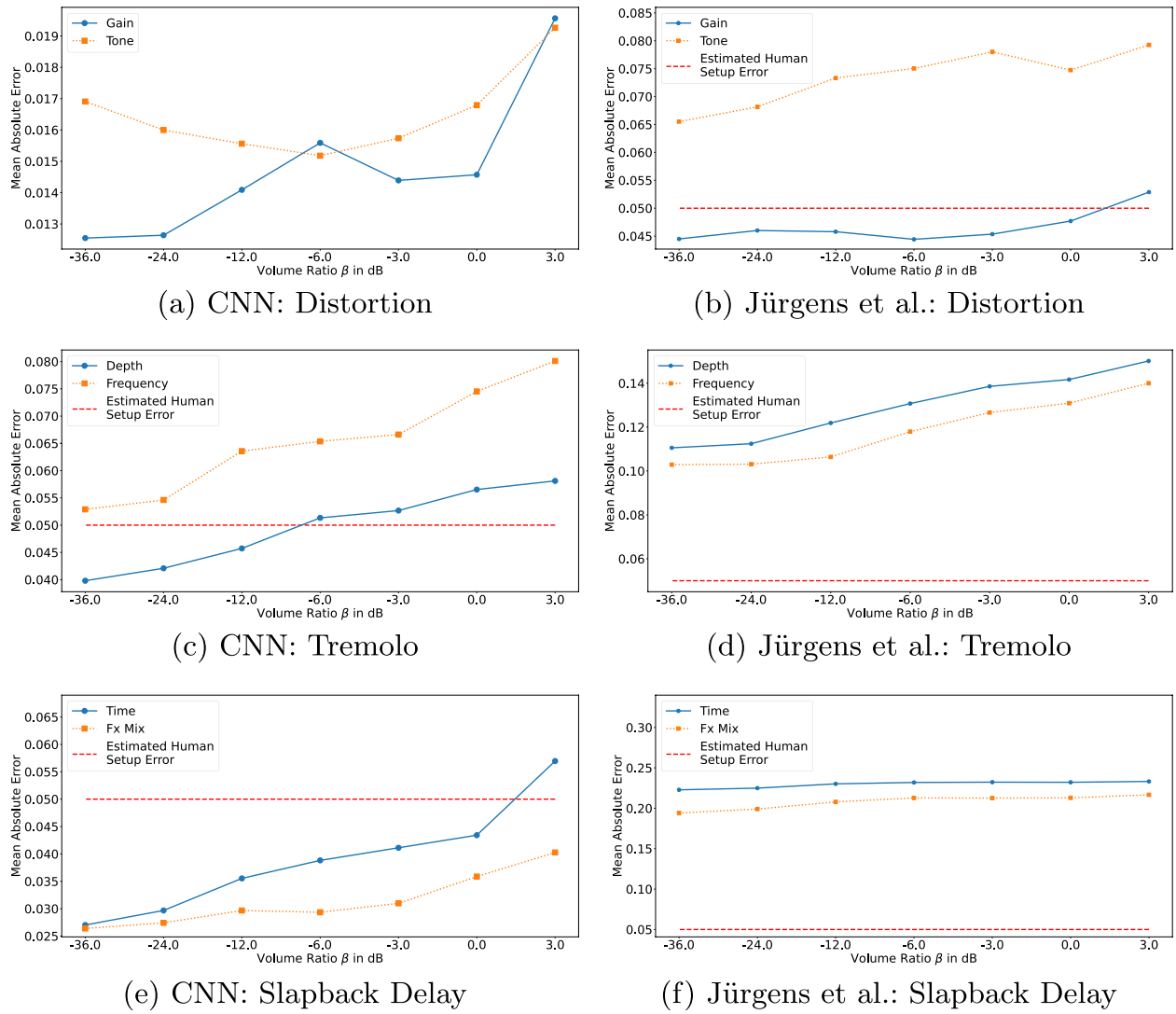


**Fig. 7** Mean absolute error of the parameter extraction of the convolutional neural network across mixing volume for the slapback delay effect and each of the four investigated time-frequency representations. Qualitatively identical results were obtained for the distortion and tremolo effect

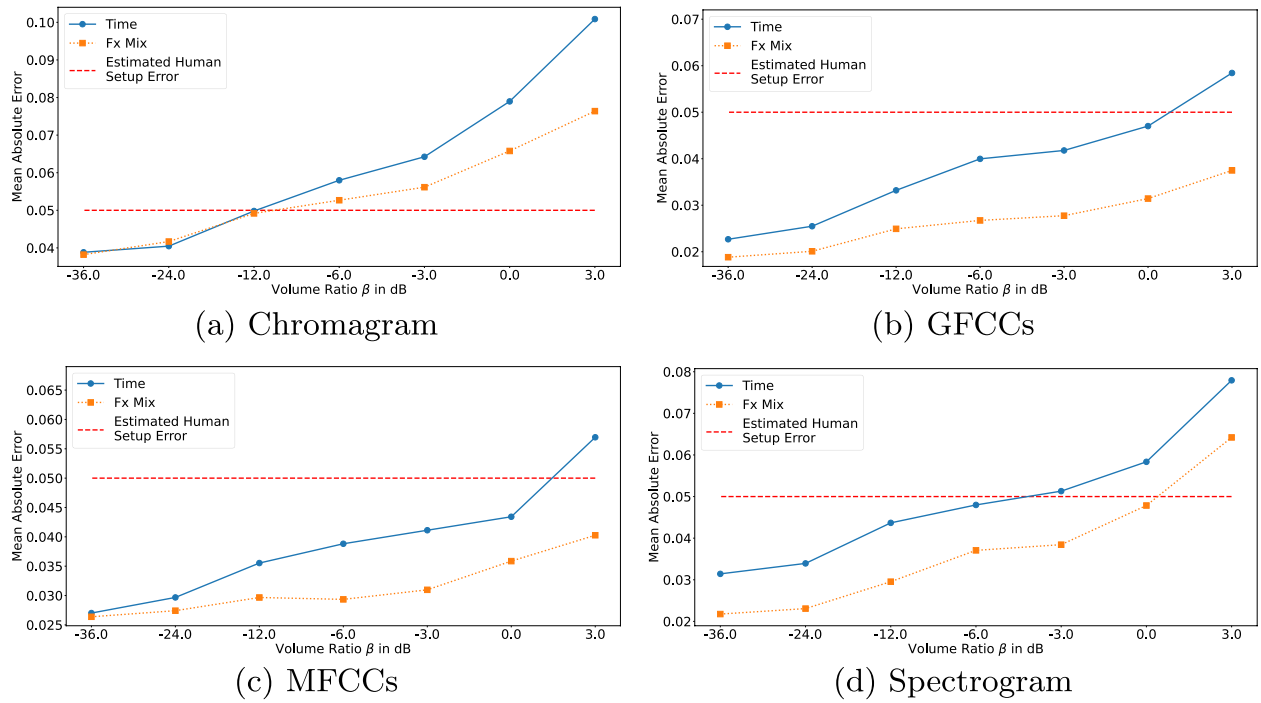


**Fig. 8** Mean absolute error of the parameter extraction for the distortion effect of the **a** convolutional neural network (CNN) and **b** Jürgens et al. [12], the tremolo effect of the **c** CNN and **d** Jürgens et al. and the slapback delay effect of the **e** CNN and **f** Jürgens et al. The CNN was used with mel-frequency cepstral coefficients as time-frequency representation

generally increased from single to multi-guitar effect extraction. For example, as a single effect using MFCCs, the setting of the depth parameter of the tremolo effect was extracted with a mean absolute error of 0.049 and rose to 0.09 when distortion, tremolo, and slapback delay were combined.

The mean absolute error across mixing volume for multi-effect parameter extraction is depicted in Fig. 10, where the spectrogram was used as a time-frequency representation. Similar results were obtained with either time-frequency representation.

### 3.4 Additional effects

Figure 11 shows boxplots of the absolute extraction errors. Table 9 gives 95 % confidence intervals of the mean absolute error of the parameter extraction for the chorus, phaser, reverb and overdrive effect. Figure 12 depicts the mean absolute error of the parameter extraction across mixing volume for the chorus, phaser, reverb, and overdrive effect. Generally, the presumed human expert setup error of 0.05 was achieved or undercut and, as with the other effects, an increase of the mean absolute error was observed for increasing mixing volume. The setting of

**Table 5** 95 % confidence intervals of the mean absolute error of the CNN across the fivefold cross-validation of the parameter extraction and all volumes. Results are reported for all time-frequency representations and the combination of distortion and tremolo effect

| **(a) Distortion + tremolo** | | | | |
|---|---|---|---|---|
| | **Gain** | **Tone** | **Depth** | **Frequency** |
| CNN + chromagram | 0.096 ± 0.002 | 0.133 ± 0.0039 | 0.094 ± 0.0024 | 0.099 ± 0.0023 |
| CNN + GFCCs | 0.077 ± 0.0007 | 0.085 ± 0.0008 | **0.056** ± 0.0034 | 0.083 ± 0.0011 |
| CNN + MFCCs | 0.079 ± 0.001 | 0.085 ± 0.003 | 0.084 ± 0.0029 | 0.097 ± 0.0039 |
| CNN + spectrogramm | **0.076** ± 0.0013 | **0.081** ± 0.0007 | 0.072 ± 0.0054 | **0.069** ± 0.0039 |

**Table 6** 95 % confidence intervals of the mean absolute error of the CNN across the fivefold cross-validation of the parameter extraction and all volumes. Results are reported for all time-frequency representations and the combination of distortion and slapback delay effect

| **(a) Distortion + slapback delay** | | | | |
|---|---|---|---|---|
| | **Gain** | **Tone** | **Time** | **Mix** |
| CNN + chromagram | 0.093 ± 0.0013 | 0.125 ± 0.0014 | 0.127 ± 0.001 | 0.082 ± 0.0013 |
| CNN + GFCCs | **0.065** ± 0.0014 | **0.07** ± 0.0019 | 0.089 ± 0.0009 | **0.051** ± 0.001 |
| CNN + MFCCs | 0.067 ± 0.0005 | 0.071 ± 0.0004 | **0.083** ± 0.0021 | 0.055 ± 0.0015 |
| CNN + spectrogramm | 0.069 ± 0.0005 | 0.074 ± 0.001 | 0.097 ± 0.0015 | 0.064 ± 0.0013 |

**Table 7** 95 % confidence intervals of the mean absolute error of the CNN across the fivefold cross-validation of the parameter extraction and all volumes. Results are reported for all time-frequency representations and the combination of tremolo and slapback delay effect

| **(a) Tremolo + slapback delay** | | | | |
|---|---|---|---|---|
| | **Depth** | **Frequency** | **Time** | **Mix** |
| CNN + chromagram | 0.106 ± 0.0017 | 0.085 ± 0.0038 | 0.152 ± 0.0024 | 0.128 ± 0.0017 |
| CNN + GFCCs | 0.071 ± 0.0018 | 0.09 ± 0.0036 | 0.106 ± 0.0029 | 0.067 ± 0.0017 |
| CNN + MFCCs | 0.092 ± 0.0034 | 0.098 ± 0.0046 | 0.084 ± 0.0017 | 0.064 ± 0.0011 |
| CNN + spectrogramm | **0.069** ± 0.0023 | **0.083** ± 0.0028 | **0.083** ± 0.0019 | **0.058** ± 0.0017 |

**Table 8** 95 % confidence intervals of the mean absolute error of the CNN across the fivefold cross-validation of the parameter extraction and all volumes. Results are reported for all time-frequency representations and the combination of distortion, tremolo and slapback delay effect

| **(a) Distortion + tremolo + slapback delay** | | | | | | |
|---|---|---|---|---|---|---|
| | **Gain** [$10^{-2}$] | **Tone** [$10^{-2}$] | **Depth** [$10^{-2}$] | **Frequency** [$10^{-2}$] | **Time** [$10^{-2}$] | **Mix** [$10^{-2}$] |
| CNN + chromagram | 7.8 ± 0.12 | 13.4 ± 0. 16 | 11.8 ± 0.27 | 10.4 ± 0.46 | 19.4 ± 0.22 | 13.5 ± 0.22 |
| CNN + GFCCs | 3.6 ± 0.06 | 4.9 ± 0.17 | **7.3** ± 0.14 | 9.1 ± 0.17 | 13.6 ± 0.18 | **8.0** ± 0.33 |
| CNN + MFCCs | 3.5 ± 0.12 | 4.5 ± 0.13 | 9.0 ± 0.3 | 10.2 ± 0.44 | **13.3** ± 0.13 | 8.6 ± 0.1 |
| CNN + spectrogram | **3.3** ± 0.014 | **4.1** ± 0.45 | 8.5 ± 0.3 | **8.6** ± 0.28 | 14.4 ± 0.12 | 10.3 ± 0.29 |

the rate parameter of the phaser effect was found to be the most difficult to extract with a mean absolute error of almost 0.06 at a mixing volume of only −12 dB.

### 3.5 Robustness to noise and pitch changes

The mean absolute error across noise levels of the effect parameter extraction is shown for all time-frequency representations in Fig. 13. Depicted are the results for the time parameter of the slapback delay. For the MFCCs and GFCCs, the mean absolute error was relatively robust to additive Gaussian noise, with an approximate increase of the mean absolute error by around 0.01–0.03 depending on the guitar effect and time-frequency representation considered. The spectrogram and chromagram both tended to be the most sensitive showing a considerable increase of about 0.05–0.15 depending on the guitar effect.

Figure 14 depicts the mean absolute error across pitch at a mixing volume of −12 dB and −3 dB for the distortion and tremolo effect. The CNN was found to be relatively robust towards unknown pitches of bass and keyboard for the distortion effect. At a mixing volume of −12 dB, all time-frequency representations, except for the chromagram, yielded mean absolute errors around or below 0.05, the presumed human expert setup error. At a mixing volume of −3 dB, all time-frequency representations except for the chromagram yielded mean absolute errors mostly between 0.05 and 0.1. The tremolo effect was less robust towards pitch shifts of keyboard and bass, with the frequency parameter being especially sensitive. An

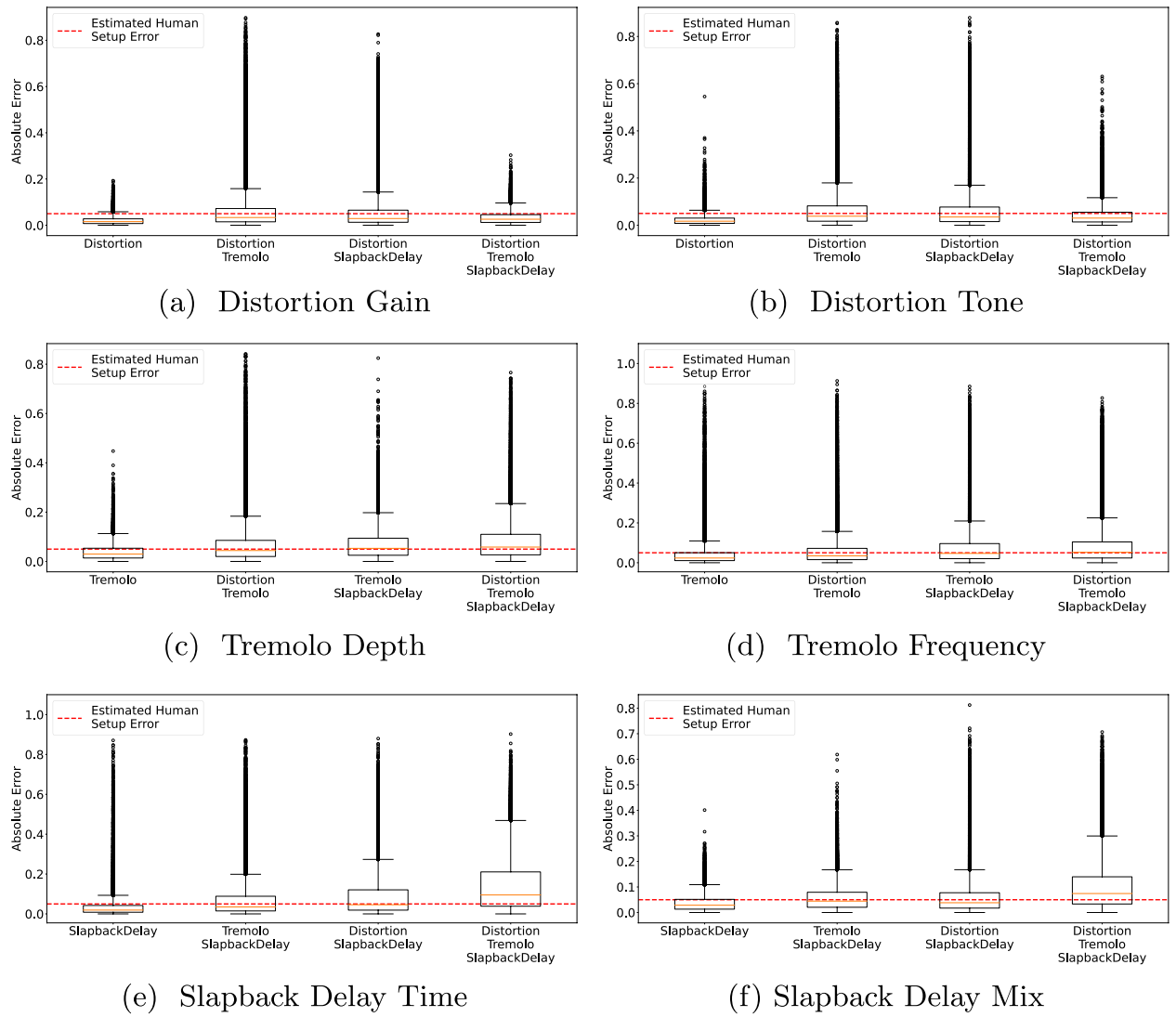


**Fig. 9** Boxplots comparing the mean absolute error of the parameter extraction with respect to the effect combination. Depicted are the absolute parameter extraction errors for the **a** gain (distortion), **b** tone (distortion), **c** depth (tremolo), **d** frequency (tremolo), **e** time (slapback delay), and **f** mix (slapback delay) parameters. These results were achieved using the spectrogram as time-frequency representation

immediate increase of the mean absolute error occurred once deviating from the outer E notes by about 0.1 at a mixing volume of −12 dB, and 0.15 at a mixing volume of −3 dB. The spectrogram, often yielding the most robust parameter extraction, yielded a considerable less sensitive parameter extraction for the depth parameter of the tremolo effect. At −12 dB, its mean absolute error was between about 0.05 and 0.15 below the mean absolute errors of the other time-frequency representations for keyboard and bass pitches between the outer E notes. At −3 dB, this gap in mean absolute error rose to about 0.1 to 0.4.

### 3.6 Error analysis

Figure 15 depicts the maximum absolute extraction error across the fivefold cross-validation for the gain parameter of the distortion effect, the frequency parameter of the tremolo effect and of the time, and the mix parameter of the slapback delay effect. These results were calculated for the MFCCs. For the gain parameter, large errors of about 0.12 almost exclusively occur for true gain settings below 0.3. In this range, the signal distortion introduced by the distortion effect occurs solely at the beginning of the audio samples. This is due to the distortion being an amplitude-dependent nonlinear effect and the natural

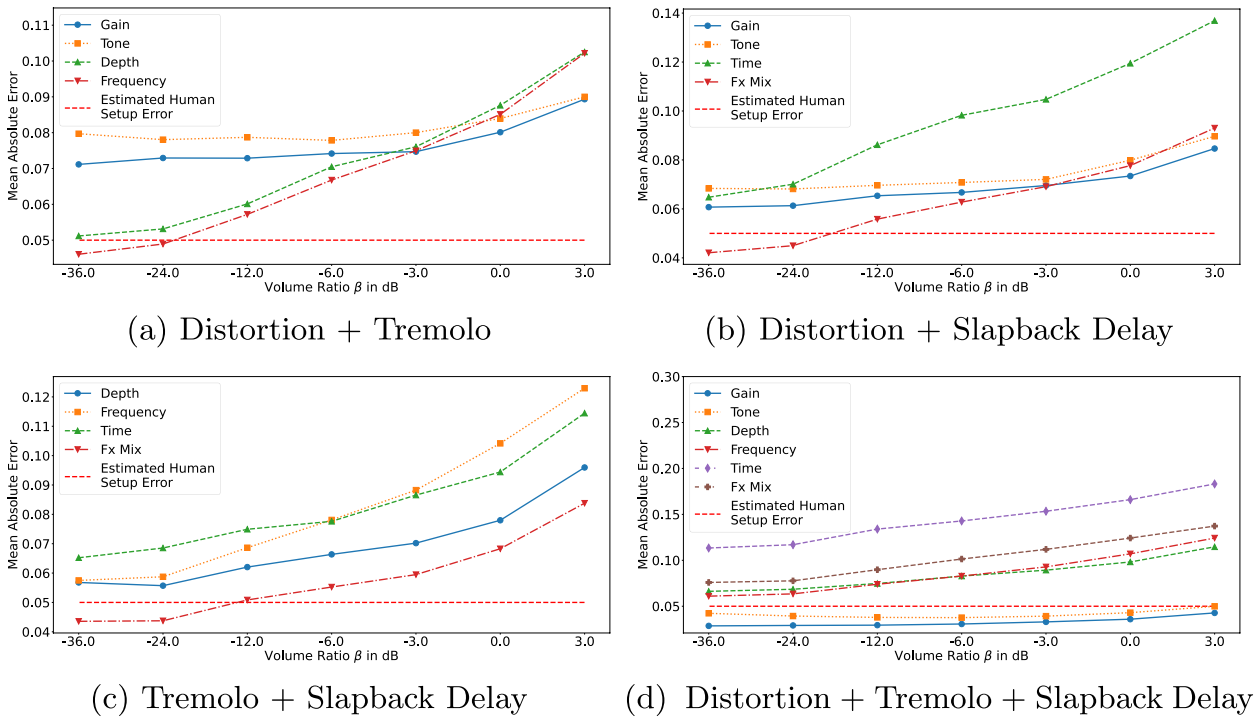


**Fig. 10** Mean absolute error of the parameter extraction across volume of the convolutional neural network (CNN) for the effect chains using the spectrogram. Depicted are **a** distortion and tremolo, **b** distortion and slapback delay, **c** tremolo and slapback delay, and **d** distortion, tremolo, and slapback delay effect

attenuation of the guitar signals. Similarly, for the time parameter of the slapback delay, the largest errors of about 0.8 occurred almost exclusively at the lowest setting of the mix parameter, i.e., when the effect was almost inaudible. For the frequency parameter of the tremolo effect, the largest errors of about 0.8 accumulated for large values of the frequency parameter in conjunction with very low values of the depth parameter. In contrast, for the mix parameter of the slapback delay, the largest errors of about 0.2 occurred for isolated parameter settings, although mildly accumulating at the largest true time parameter setting.

## 4 Discussion

Generally, the CNN was found to be superior to the investigated baselines in both effect classification and effect parameter extraction. All time-frequency representations allowed to achieve an accuracy around or better than the human expert level in parameter extraction, with the chromagram performing the worst out of the four. MFCCs and GFCCs were found to perform approximately the same, neither having a clear edge over the other.

### 4.1 Effect classification

The CNN outperformed the SVM classifier on both datasets, except when using the chromagram on the IDMT-SMT and GFCCs on the GEC-GIM dataset. The chromagram contains the most coarse information about the audio out of all investigated time-frequency representations, as it maps different octaves to the same pitch value. Therefore, it performing the worst was not surprising. Interestingly, the CNN achieved, in its best configuration, an accuracy of 97.4 % on the IDMT-SMT which is very close to the 97.7 % reported by Stein et al. [4] in their SVM implementation for monophonic guitar samples. Albeit they included, in contrast to our work, the unprocessed guitar signals as an additional effect class, this may be indicative of an upper bound of the achievable classification accuracy on the IDMT-SMT. As the CNN achieved good results for both datasets, one using virtual, sample-based instruments, with all kinds of instrument mixes, and the other real recordings and a large variety of solo guitar pitches, it can be assumed that the CNN is suitable for a wide range of audio data. The confusion of slapback and feedback

delay on the GEC-GIM was due to the feedback delay being very difficult to make out for settings at least below 0.3, confirmed informally through selected listening tests. Therefore, in about 30 % of cases due to the random settings used, the slapback delay and feedback delay samples were in large parts indeed very difficult to distinguish, and the confusion of these two effects by the CNN is no indicator of poor performance. The CNN and the SVM both failing to distinguish these two effects on the GEC-GIM dataset, and both not failing to do so on the IDMT-SMT dataset, is a strong indicator that the issue lies in the data and not the classifiers.

**Table 9** 95 %gives 95 % confidence intervals of the mean absolute error of the parameter extraction of the convolutional neural network for the chorus, phaser, reverb, and overdrive effect using mel-frequency cepstral coefficients as time-frequency representation

| Effect | Parameter 1 | Parameter 2 |
|---|---|---|
| Chorus (Rate. Depth) | $0.025 \pm 0.0006$ | $0.035 \pm 0.0007$ |
| Phaser (Rate. Depth) | $0.058 \pm 0.0009$ | $0.031 \pm 0.0023$ |
| Reverb (Room Size. Mix) | $0.041 \pm 0.0009$ | $0.034 \pm 0.0006$ |
| Overdrive (Gain. Tone) | $0.024 \pm 0.0007$ | $0.029 \pm 0.0016$ |

### 4.2 Parameter extraction

The impact of the mixing volume on the parameter extraction error qualitatively was as expected, with the error increasing generally with increasing mixing volume. For single effects, MFCCs and GFCCs performed the best, the spectrogram achieving similar but worse performance. For multi-effects, except for the chromagram, which performed the worst, no time-frequency representation was consistently better than the others. However, considering the impact of the time-frequency representations on the number of parameters of the CNNs, the MFCCs outperformed all others and are commended for future work.

For all effects and volumes, the CNN outperformed the approach of Jürgens et al. [12], although the CNN showed a greater sensitivity towards the mixing volume. While the boxplots revealed considerable outliers, usually about 75% or more of the extraction errors were below the error of a human expert. Outliers, as suggested by Fig. 15, usually occurred at settings that were very difficult to distinguish and also at high mixing volumes. Some parameters have an impact on each other and render the other virtually impactless at certain settings. An example is the mix parameter of the slapback delay effect, which renders the time parameter subjectively impactless if set very close to zero. Then, parameter extraction

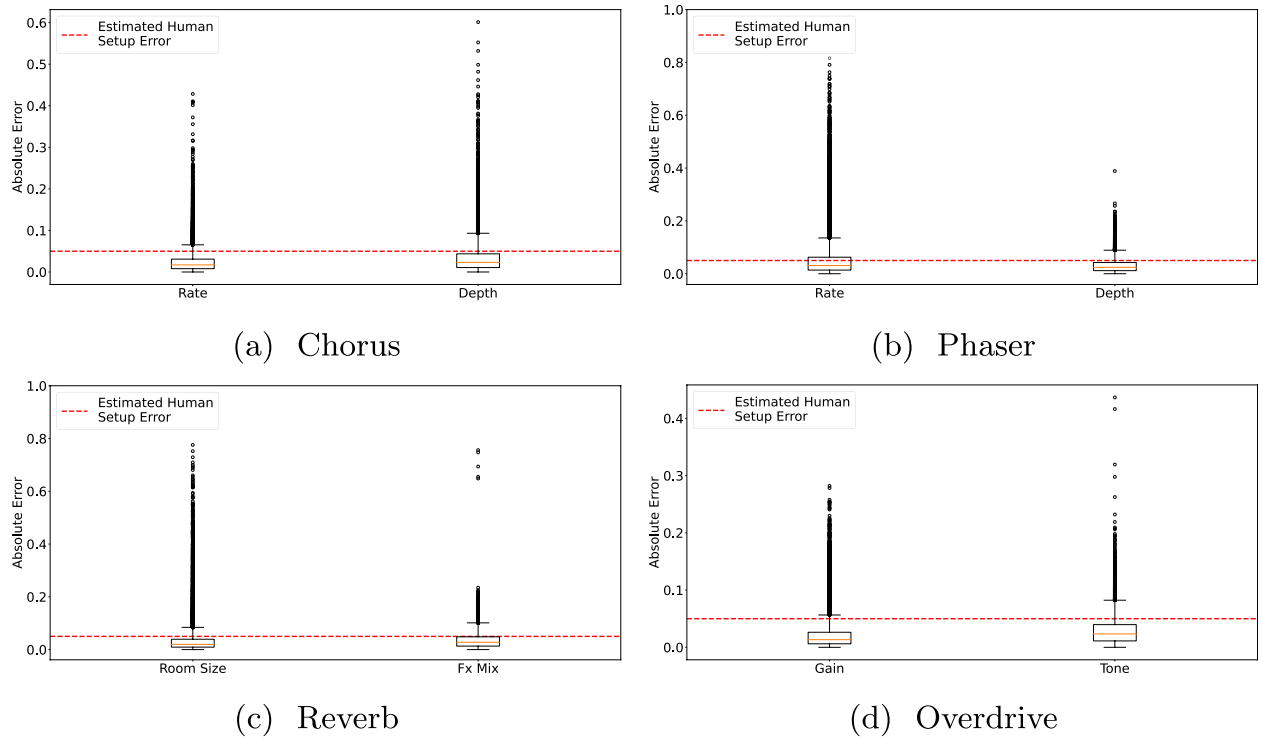


**Fig. 11** Boxplots of the absolute error of the parameter extraction using the convolutional neural network with mel-frequency cepstral coefficients for the **a** chorus, **b** phaser, **c** reverb, and **d** overdrive effect

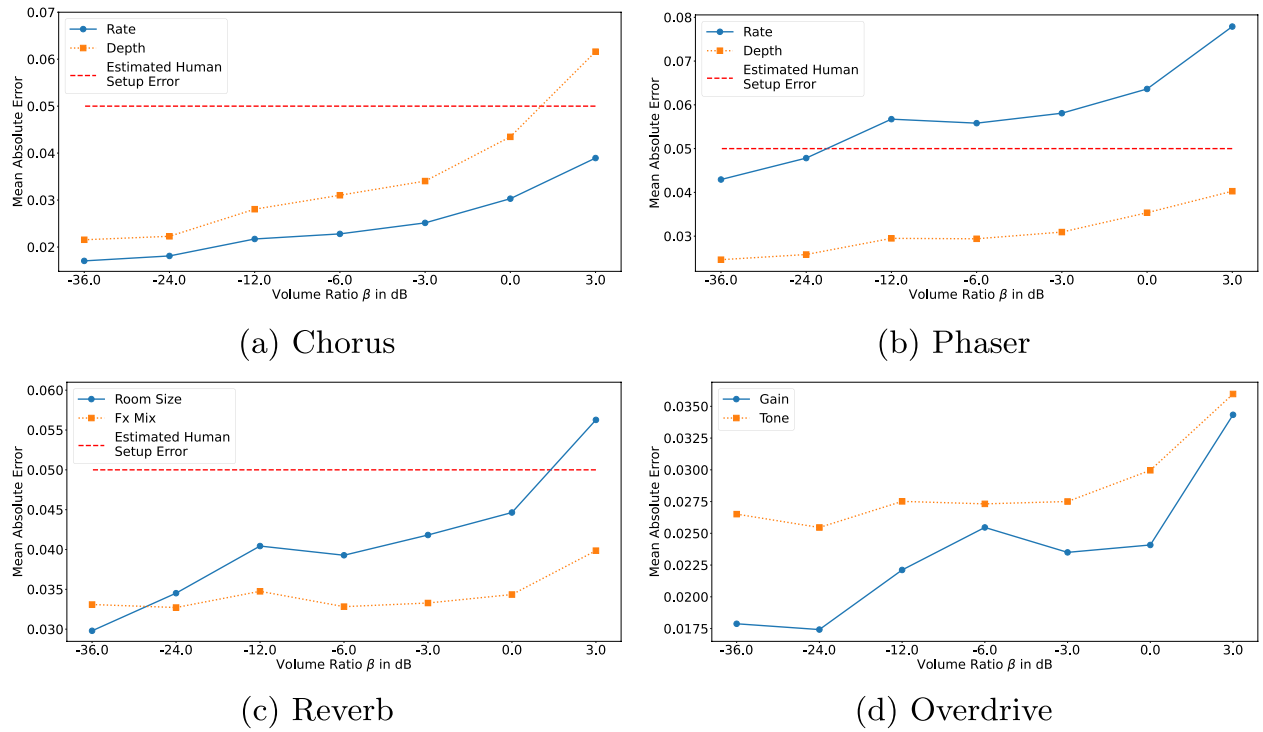


**Fig. 12** Mean absolute error of the parameter extraction across mixing volume using mel-frequency cepstral coefficients for the **a** chorus, **b** phaser, **c** reverb, and **d** overdrive effect. The presumed human expert setup error of 0.05 was added as well, where appropriate, as a reference

is considerably more difficult. Nonetheless, a few isolated large errors occur at apparently random parameter settings without an obvious explanation and also intermediate mixing volumes. These large errors perhaps indicate an insufficient amount of data or suboptimal data creation, i.e., more data is needed around critical parameter settings. However, if we apply the accuracy metric used by Comunità et al. [13], the publication most similar to this work, which considered root-mean-square absolute errors of the parameter settings below 0.1 as correct, in almost all conditions, the CNN correctly extracted the parameter settings. Specifically for the distortion effect, some comparisons can be made to the results by Comunità et al., which only investigated nonlinear effects. For their discrete monophonic dataset, they report a mean absolute extraction error of 0.03 for the gain and 0.039 for the tone parameter. This is about twice the error achieved in this work; however, Comunità et al. did not consider instrument mixes, but extraction of parameter settings from different implementations of the same guitar effect. However, they saw an improvement when training with polyphonic guitar samples, achieving mean absolute extraction errors down to around 0.02, which is very similar to our performance. They argued this was due to richer information being present in polyphonic recordings. Similarly, our low extraction errors could be due to considering richer information through the different mixing volumes.

Interestingly, except for the distortion effect, the absolute extraction error increased with the increasing number of guitar effects. The mean absolute error of the parameter extraction based on the spectrogram increased the most for the slapback delay effect, from a mean absolute error of 0.049 for the individual effect to 0.144 when distortion, tremolo, and slapback delay were combined. However, the slapback delay was also the last effect applied in the effect chain made of distortion, tremolo, and slapback delay. Similarly, the tremolo effect was the second effect applied and did not see an increase as sharp, with a mean absolute error for the frequency parameter of, e.g., 0.052 for the individual effect and 0.086 for the distortion, tremolo, and slapback delay effect chain. No consistent increase was observed for the distortion effect. This suggests that the increase in extraction error might be linked to the position in the effect chain.

Initially, we investigated, whether training on instrument mixes was necessary to achieve high performance. One could train the CNNs on unmixed guitar signals, to then apply the CNNs on instrument mixes. However, as the results were poor even at intermediate mixing volumes — classification accuracy dropped to below

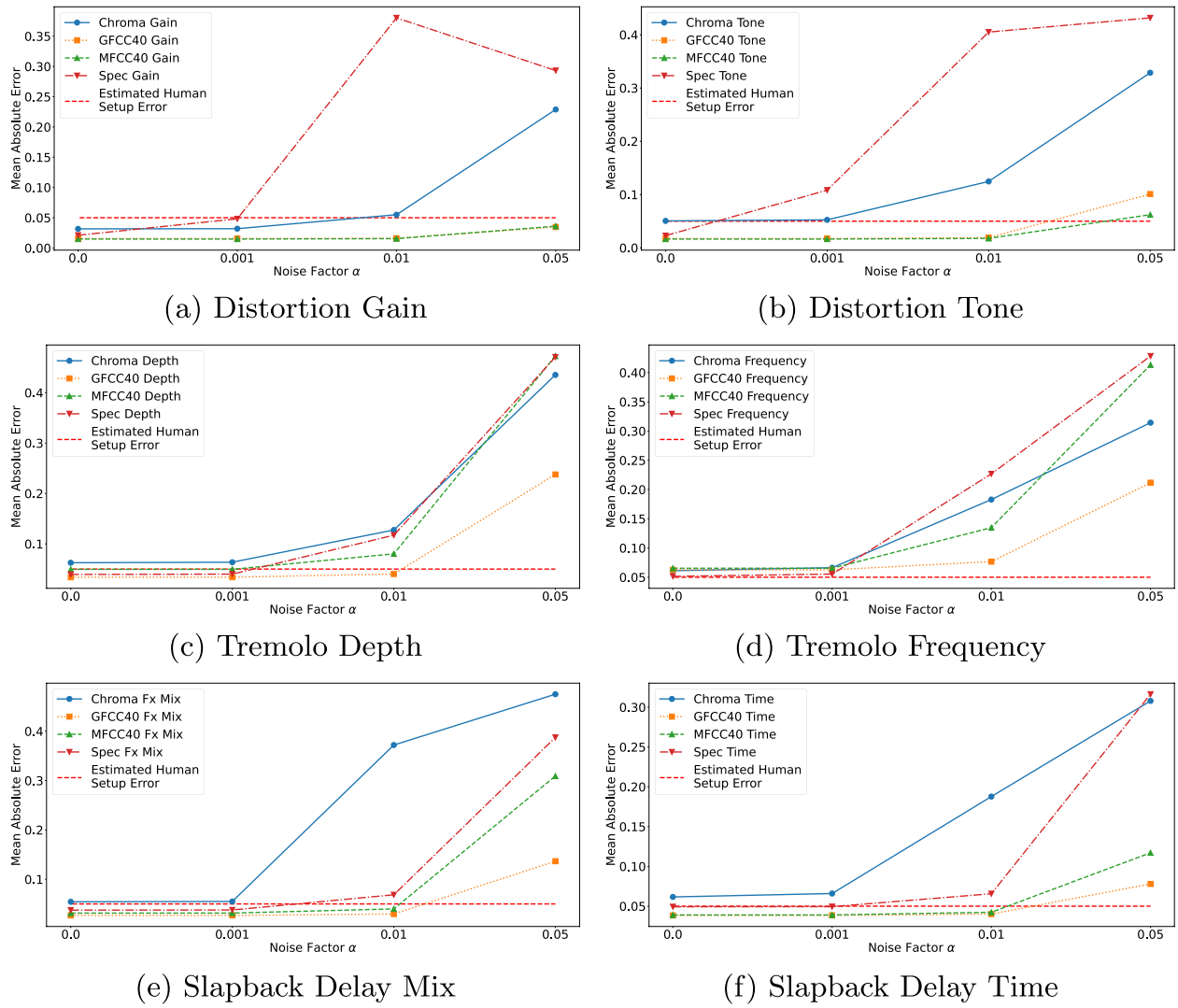


**Fig. 13** Mean absolute error of the parameter extraction of the convolutional neural network across background noise levels for the **a**, **b** distortion, **c**, **d** tremolo, and **e**, **f** slapback delay effect and all investigated time-frequency representations. The noise factor $\alpha$ was described in Section 2.3.2

40% when only a kick drum was mixed with the guitar, it became clear that this was not a reasonable approach. Another point that could be raised is that source separation algorithms could be applied, to separate the guitar signal from the other instruments first and to then apply the parameter extraction/CNN. However, in pilot investigations using FASST [27], we noticed that FASST did not separate the instrument mix into the separate instruments. It rather tended to separate sources present in more than one channel from those that are not. Considerable artifacts from other instruments, e.g., drums, remained and would deteriorate algorithms that assume unmixed guitar signals. Other, newer source separation algorithms might be beneficial, but after these initial results, we focused on direct extraction of the parameter settings.

A drawback of the proposed method to parameter extraction is the necessity to train specialized neural networks for any effect and any effect combination considered. It is very likely that different implementations of the same effect, say phaser, while obviously belonging to the same effect type to the human ear, would yield very poor results with a CNN trained on another implementation. Additionally, due to combinatoric explosion, even if the CNNs were generalizing to different implementations, the vast number of possible effect combinations would yield our approach computationally too expensive. Therefore, a technique or approach

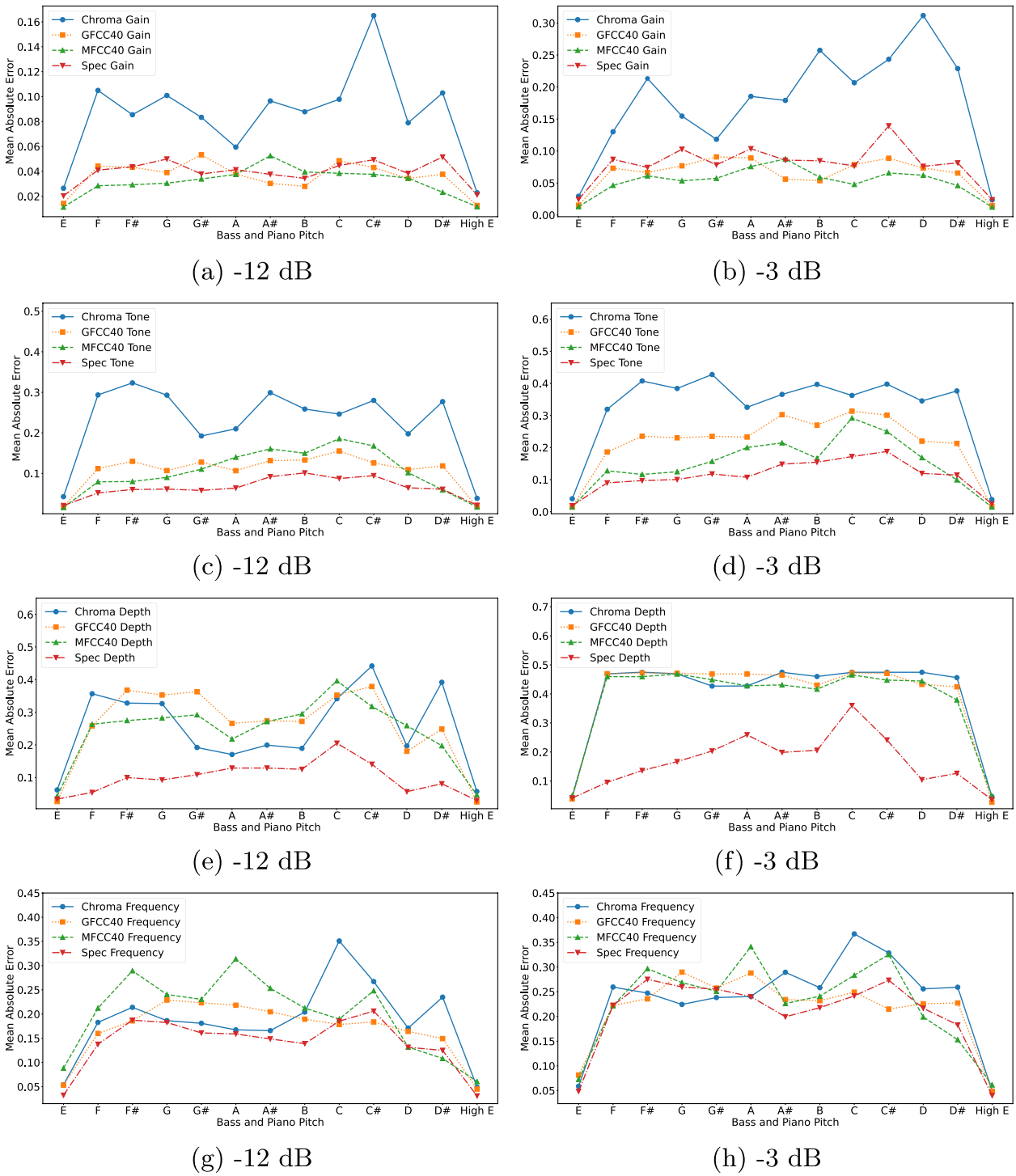


**Fig. 14** Mean absolute error of the parameter extraction of the convolutional neural network across pitch of keyboard and bass for all time-frequency representations investigated. Only the two E tones were part of the training. Shown are, respectively, the extraction errors for the gain, tone, depth, and frequency parameter of the distortion and tremolo effect at −12 dB mixing volume (**a**, **c**, **e**, **g**) and at −3 dB mixing volume (**b**, **d**, **f**, **h**)

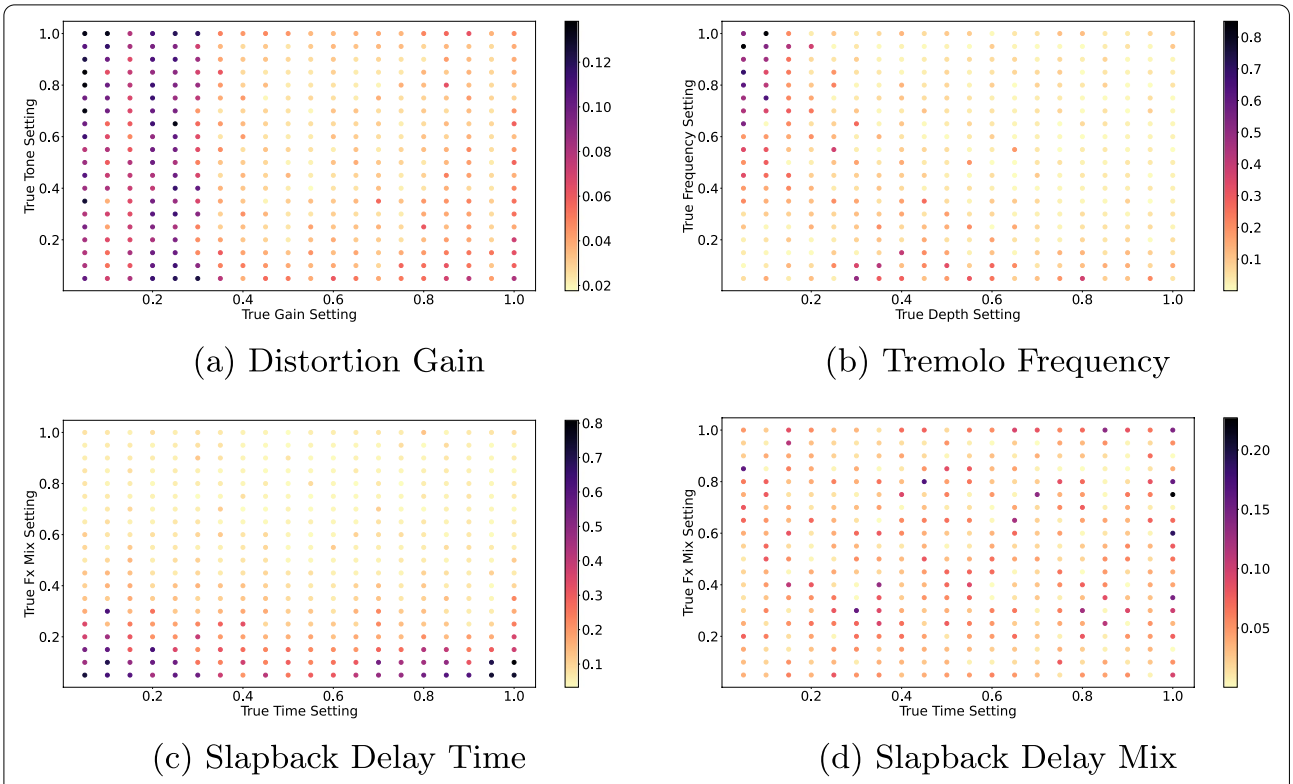


**Fig. 15** Maximum of the absolute parameter extraction error for the **a** distortion gain, **b** tremolo frequency, **c** slapback delay time, and **d** slapback delay mix parameter across the ground truth parameter settings. The maximum was taken across the fivefold cross validation

to generalize to effect combinations should be investigated in the future. To verify that this was an issue, we used a CNN trained on single effect samples using the distortion effect and tested it on multi-effect samples using the distortion and tremolo effect. Using the spectrogram, this CNN achieved a mean absolute extraction error of 0.021 and 0.023 for the gain and tone parameters, respectively. This CNN achieved a mean absolute extraction error of 0.119 and 0.126 for the gain and tone parameters, respectively, on the multi-effect samples. This is in contrast to mean absolute extraction errors of 0.076 and 0.081 for the gain and tone parameters, respectively, when the CNN was trained specifically on this effect combination, an increase of about 50 %. As distortion was generally the easiest effect to extract, this suggests that specifically trained CNNs are indeed required to achieve high performance on multi-effect samples.

### 4.3 Robustness

As the CNN showed only a minor decrease in accuracy when small amounts of noise were added to the input time-frequency representations, over-fitting, at least to the individual pixels, seems unlikely. Some deep networks are susceptible to these types of over-fitting [16]. Because at lower mixing volumes robustness was also observed across pitch, it is very probable that the CNN indeed extracts meaningful features. Albeit at the largest noise level the noise was considerable, it was estimated that a human expert would not see a similarly large increase in parameter extraction error as the CNN. Here, the CNN potentially performed suboptimally and could be improved in the future. Improvements could be achieved by inclusion of noisy samples in the dataset. The chromagram generally yielding the least robust CNN is due to the fact that the energy for the chromagram in the training data was almost solely contained in the E note. The CNN then had to focus mostly on this particular pitch. Once the other instruments changed their pitch, and therefore the energy distribution, this approach was prone to fail.

The extraction error for the gain parameter of the distortion effect — and less so the tone parameter — was found to be rather robust to pitch changes of keyboard and bass. The CNN achieved, except when using the chromagram, around and below 0.1 mean absolute error even at −3 dB mixing volume, unlike all other effects and parameters. The reason likely is

the nonlinear nature of the distortion effect. As such, it introduces novel frequencies into the audio signals, which certainly convey information about the parameter settings. Therefore, it appears reasonable that the CNN learned a more diverse look at the time-frequency representations for the distortion effect. In return, this allowed to be more robust to changes of the pitch of the other instruments. Although the number of weights of the CNN, especially for the spectrogram, was rather large in comparison to the amount of training data, the results of the robustness analysis and the use of dropout layers make it seem unlikely that relevant over-fitting occurred.

### 4.4 Limitations and future work

One limitation of our investigation is the simplicity of the music played by the instruments. More complex musical pieces, including polyphonic guitar melodies, likely will be more challenging to extract guitar effects from. The true variability of recorded music, like different guitar timbres or further audio tracks, should be focused on in the future. Furthermore, additional audio effects on the other instruments could interfere with the parameter extraction of the guitar effects. Also, a second guitar could have a considerable impact on the classification and extraction performance. Finally, a thorough subjective listening test would be desirable to precisely measure the human setup performance.

## 5 Conclusion

In this work, guitar effect classification and guitar effect parameter extraction with convolutional neural networks (CNNs) from instrument mixes was investigated and compared to two baseline approaches. Four time-frequency representations were investigated, namely the spectrogram, mel-frequency cepstral coefficients, gammatone-frequency cepstral coefficients, and the chromagram.

On two datasets, the CNN achieved classification accuracies $1-5\%$ above the baseline accuracy, achieving up to 97.4 % accuracy. Mean absolute parameter extraction errors of below 0.016 for the distortion, below 0.052 for the tremolo, and below 0.038 for the slapback delay effect were achieved, matching or surpassing the presumed human expert error of 0.05. The CNN approach was found to generalize to further effects, achieving mean absolute parameter extraction errors below 0.05 for the chorus, phaser, reverb, and overdrive effect. For sequentially applied combinations of distortion, tremolo, and slapback delay, the mean extraction error slightly increased from the performance for the single effects to the range of 0.05 to 0.1. The CNN was found to be moderately robust to noise and pitch changes of the background instrumentation suggesting that the CNN extracted meaningful features. Due to their high performance at a moderate number of CNN parameters, MFCCs are recommended for future work.

**Acknowledgements**
N/A.

**Authors' contributions**
RH: main idea, design of experiments and evaluations, and author of this manuscript. KG: implementation, visualization, and proof-reading. AL: proof-reading. JO: supervision and proof-reading. The authors read and approved the final manuscript.

**Funding**
Open Access funding enabled and organized by Projekt DEAL. The research has not been funded by third parties.

**Availability of data and materials**
The code can be found under https://github.com/kevingerkens/gitfx and the datasets under https://seafile.cloud.uni-hannover.de/d/5398a844db214b5fb31b/. Audio samples showcasing the parameter extraction performance of the CNN and additional audio examples can be found under https://seafile.cloud.uni-hannover.de/d/115de52e9a574882a152/.

## Declarations

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

Received: 21 March 2022 Accepted: 23 August 2022
Published online: 23 October 2022

**References**

1. T. Wilmering, D. Moffat, A. Milo, M. Sandler, A history of audio effects. Appl. Sci. **10**, 791 (2020)
2. A. Sarti, U. Zoelzer, X. Serra, M. Sandler, S. Godsill, Digital audio effects. EURASIP J. Adv. Signal Proc. **2010**(1), 459654 (2011). https://doi.org/10.1155/2010/459654
3. U. Zölzer, *DAFX: Digital Audio Effects*, 2nd edn. (Wiley, Chichester, 2011)
4. M. Stein, J. Abeßer, C. Dittmar, G. Schuller, *Automatic detection of audio effects in guitar and bass recordings* (J. Audio Eng, Soc, 2010)
5. M. Stein, in Proc. of the 13th International Conference on Digital Audio Effects (DAFx 2010). Automatic detection of multiple, cascaded audio effects in guitar recordings (2010)
6. F. Eichas, M. Fink, U. Zölzer, in Proceedings of the 18th International Conference on Digital Audio Effects (DAFx-15). Feature design for the classification of audio effect units by input / output measurements (2015)
7. M. Schmitt, B. Schuller, Recognising guitar effects - which acoustic features really matter?, in *INFORMATIK 2017*. (Gesellschaft für Informatik, Bonn, 2017), pp.177–190
8. N. Masuda, D. Saito, in Proceedings of the 24rd International Conference on Digital Audio Effects (DAFx2021). Quality diversity for synthesizer sound matching (2021)
9. M.J. Yee-King, L. Fedden, M. d'Inverno, Automatic programming of VST sound synthesizers using deep networks and other techniques. IEEE Trans. Emerg. Top. Comput. Intell. **2**(2), 150–159 (2018)
10. D. Sheng, G. Fazekas, in Proceedings of the 20rd International Conference on Digital Audio Effects (DAFx2017). Automatic control of the dynamic range compressor using a regression model and a reference sound (2017)
11. D. Sheng, G. Fazekas, in 2019 International Joint Conference on Neural Networks (IJCNN). A feature learning siamese model for intelligent control of the dynamic range compressor (IEEE, 2019), p 1–8

12. H. Jürgens, R. Hinrichs, J. Ostermann, in Proceedings of the 23rd International Conference on Digital Audio Effects (DAFx2020). Recognizing guitar effects and their parameter settings (2020)
13. M. Comunitá, D. Stowell, J.D. Reiss, Guitar effects recognition and parameter estimation with convolutional neural networks. J. Audio Eng. Soc. **69**(7/8), 594–604 (2021)
14. R. Hinrichs, K. Gerkens, J. Ostermann, in 11th International Conference on Artificial Intelligence in Music, Sound, Art and Design (EvoMUSART). Classification of guitar effects and extraction of their parameter settings from instrument mixes using convolutional neural networks (2022). https://doi.org/10.1007/978-3-031-03789-4_7
15. A. Nguyen, J. Yosinski, J. Clune, in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images (2015), p 427–436
16. J. Su, D.V. Vargas, K. Sakurai, One pixel attack for fooling deep neural networks. IEEE Trans. Evol. Comput. **23**(5), 828–841 (2019)
17. M. Cogswell, F. Ahmed, R. Girshick, L. Zitnick, D. Batra, Reducing Overfitting in Deep Networks by Decorrelating Representations (2016)
18. Kjaerhus Audio: Kjaerhus Audio Classic Series. Website. https://vstwarehouse.com/d/kjaerhus-audio-classic-series
19. SimulAnalog: SimulAnalog Guitar Suite. Website. https://www.simulanalog.org/guitarsuite.htm
20. Pechenegfx: Pecheneg Tremolo. Website. http://pechenegfx.org/plugins/pecheneg-tremolo/
21. Buzzroom: OctBUZ. Website. https://plugins4free.com/plugin/582/
22. M. Müller, *Fundamentals of Music Processing: Using Python and Jupyter Notebooks* (Springer, Cham, 2021)
23. M. Jeevan, A. Dhingra, M. Hanmandlu, B. Panigrahi, Lecture notes in electrical engineering. Robust speaker verification using GFCC based i-vectors, in *Proceedings of the International Conference on Signal, Networks, Computing, and Systems*. (Springer, New Delhi, 2017)
24. S. Chaudhary, S. Kakarwal, R. Deshmukh, Musical instrument recognition using audio features with integrated entropy method. J. Integr. Sci. Technol. **9**(2), 92–97 (2021). http://www.pubs.iscience.in/journal/index.php/jist/article/view/1359
25. G.K. Liu, Evaluating Gammatone Frequency Cepstral Coefficients with Neural Networks for Emotion Recognition from Speech. arXiv preprint arXiv:1806.09010. (2018)
26. B. Carter, S. Jain, J. Mueller, D. Gifford, Overinterpretation reveals image classification model pathologies. arXiv preprint arXiv:2003.08907. (2021)
27. Y. Salaün, E. Vincent, N. Bertin, N. Souviraá-Labastie, X. Jaureguiberry, D.T. Tran, et al., in ICASSP 2014; 2014. The flexible audio source separation toolbox version 2.0


## Publisher's Note