Recognising Guitar Effects – Which Acoustic Features Really Matter?

Maximilian Schmitt[1] Björn Schuller[2]

Abstract: The recognition of audio effects employed in recordings of electric guitar or bass has a wide range of applications in music information retrieval. It is meaningful in holistic automatic music transcription and annotation approaches for, e. g., music education, intelligent music search, or musicology. In this contribution, we investigate the relevance of a large variety of state-of-the-art acoustic features for the task of automatic guitar effect recognition. The usage of functionals, i. e., statistics such as moments and percentiles, is hereby compared to the bag-of-audio-words approach to obtain an acoustic representation of a recording on instance level. Our results are based on a database of more than 50 000 monophonic and polyphonic samples of electric guitars and bass guitars, processed with 10 different digital audio effects.

Keywords: Guitar Effects; Music Information Retrieval; Bag-of-Audio-Words

1 Introduction

Digital audio effects are methods of modifying the acoustic waveform in audio (mostly music) recordings. Applying audio effects can make a recording more vivid, change or enhance the mood of a musical piece, or simulate the sound of arbitrary instruments or environments. During the last century, a large number of effects have evolved. In contrast to the first realisations of acoustic effects, which were based on electromechanical devices or analogue circuits, the majority of nowadays' effects are implemented in digital circuits or in software, as part of a digital audio workstation (DAW). These provide the artist with a large bandwidth of methods to modify, enhance, or mix audio recordings.

While the generation of digital audio effects is well-studied [Zö11], this is not the case for their automatic recognition. However, methods to automatically detect and classify audio effects applied to a signal have several applications in the field of *music information retrieval* (MIR). First of all, it can be employed to enhance the performance of *automatic music transcription* systems [KV09, Be13, Ke14]. In this context, effect recognition can be used in two ways: Firstly, recognising the acoustic transformation of the signal prior to or in combination with pitch detection can improve its accuracy. Secondly, providing

[1] Chair of Complex and Intelligent Systems, Universität Passau, Innstraße 43, 94032 Passau, Germany
maximilian.schmitt@uni-passau.de

[2] Chair of Complex and Intelligent Systems, Universität Passau, Innstraße 43, 94032 Passau, Germany;
Department of Computing, Imperial College London, 180 Queens' Gate, London SW7 2AZ, U. K.
schuller@ieee.org

information about instrumentation and applied audio effects themselves to the user, can augment automatic music transcription or annotation systems used for music education [Di12] and musicology [Ab17]. Moreover, the analysis of employed effects can be an important aspect in *intelligent music search* and *recommendation* applications, as some effects might be related to a certain mood or genre the user is looking for [SDP12].

Previous research on automatic audio effect recognition in guitar and bass recordings has been conducted by Stein et al. [St10b]. The authors created the IDMT-SMT-Audio-Effects database[3] of monophonic and polyphonic guitar recordings and monophonic bass recordings. Isolated tones or chords have been recorded and modified with 10 different effects in a DAW. The authors extracted three types of acoustic (spectral, cepstral, and harmonic) features on the sustain part of each sample and employed a *support vector machine* for classification. In addition to the isolated samples, the evaluation was done on audio mixtures of multiple instruments. A similar approach has also been used by Stein for the recognition of cascaded effects [St10a].

In this contribution, we pursue an approach which does not incorporate any preprocessing or segmentation of the samples. Thus, the attack and decay parts of the notes, which are highly depending on the plucking style and the instrument itself, are present in the analysed audio. This makes the whole task more challenging for a system that is supposed to work independent from an instrument and the playing style, on the one hand, but on the other hand, it is not depending on any preprocessing, such as onset detection, which can introduce additional error. We compare the performance of a recognition system based on different classes of standard acoustic features, combined on instance level using either functionals (statistics, such as *moments* of different orders, *percentiles*, and *extrema*) or a *bag-of-audio-words* (BoAW) representation.

In the BoAW method, an audio sample is summarised as a histogram of vector quantised frame-level acoustic features. The approach has been introduced originally in the field of *natural language processing*, where it is known under the name *bag-of-words*, but has gained increasing interest in the visual (*bag-of-visual-words, BoVW*) and audio community. Most of the research on BoAW has been done on *acoustic event detection* and *classification* [PA12, Ra13, GPF15], but it has also been applied successfully in the context of further machine recognition tasks, such as *emotion recognition* in speech [Po15, SRS16] and classification of *snore sounds* [Sc16]. Also in the field of MIR, a number of publications on BoAW already exists. Riley et al. exploit the approach for *audio fingerprinting*, based on *Chroma* features [RHG08]. Authors report that their method works well for the detection of cover/live versions of songs, even if the cover is much longer due to an extended guitar solo. This shows that BoAW can capture the overall distribution of features and match versions of songs as long as the distribution of chords or musical keys is similar. Yeh et al. used BoAW based on the short-time spectrum for *music genre classification* [YSY13].

[3] Link to the IDMT-SMT-Audio-Effects database: `https://www.idmt.fraunhofer.de/en/business_units/m2d/smt/audio_effects.html`

This contribution is organised as follows: After a description of the IDMT-SMT-Audio-Effects database in Section 2, we explain our approach in detail in Section 3. Experiments and results are presented in Section 4 and discussed in Section 5. In Section 6, we conclude and give an outlook on future research in the field.

2 Database

The IDMT-SMT-Audio-Effects database by Stein et al. [St10b] contains isolated monophonic and polyphonic samples of guitars and monophonic samples of bass guitars. In detail, the following four instruments have been recorded, each one in two different configurations:

1. Yamaha BB604 (bass)
2. Warwick Corvette (bass)
3. Schecter Diamond C-1 Classic (guitar)
4. Chester Stratocaster (guitar)

For the monophonic recordings, each note between the 0-th and the 12-th fret has been recorded in two different plucking styles (finger and plectrum). For the polyphonic database, several chords (consisting of 2 to 6 notes) have been recorded for each of the two guitars using a plectrum. Each sample has a duration of approximately 2 seconds and was modified in a DAW using 10 different audio effects. The effects can be categorised into 3 groups as shown in Table 1.

The *modulating* effects are those effects where the basic parameters of the approximately sinusoidal audio signal (amplitude, frequency, and phase) are varied over time [Zö11]; whereas *Tremolo* denotes an amplitude modulation and *Vibrato* denotes a frequency modulation[4]; the effects *Chorus*, *Flanger*, and *Phaser* mix the audio signal with a shifted version of itself, with a time-varying delay. For *Chorus*, this is done in a rather random manner, simulating the presence of several synchronous instruments (‘choir’) in a single-instrument track. For *Flanger* and *Phaser*, the delay is given by a sinusoid, whose frequency is lower for the *Phaser*.

Ambience effects try to simulate different environments. *Reverb* adds a quite natural reverberation to the sound as present in recording rooms in case they are not anechoic. *Delay* adds a distinguishable copy of each sound after an interval of around 0.1 s to 2.0 s, so that it is perceived as an ‘echo’. While *Slapback Delay* manifests only as a single copy, *Feedback Delay* generates multiple copies of the signal.

Finally, guitar or bass recordings are very often subject to *nonlinear* distortions. Those effects are usually generated already through the nonlinear characteristics of the amplifying

[4] In practice, *Vibrato*, especially in the singing voice, comes also with an amplitude modulation due to physiological constraints.

Effect group		
Modulation	**Ambience**	**Nonlinear distortions**
Chorus	Reverb	Overdrive
Flanger	Feedback Delay	Distortion
Phaser	Slapback Delay	
Tremolo		
Vibrato		

Tab. 1: Audio effects present in the IDMT-SMT-Audio-Effects database (10 audio effects in 3 groups).

elements (tube or transistor) in a guitar or bass amplifier. Nevertheless, the effects can be simulated in a DAW. The difference between *Overdrive* and *Distortion* is mainly defined by the operating point on the characteristic curve. While *Overdrive* specifies an operating range in both the linear and the nonlinear area, for the more intense effect of *Distortion*, only the nonlinear part is used [Zö11].

In addition to the described audio effects, the unprocessed audio samples (noFX) are contained in the database. While for the processed samples, three different effect settings are available for each sample, the samples without effects are augmented with two different amplifier simulations, resulting in the same number of samples for each effect and the noFX class[5]. This leads to a database of 55 044 samples; 10 296 monophonic instances for each instrument and 6 930 instances for the polyphonic recordings of each guitar.

3 Methodology

The approach proposed in this section consists of three steps. First, acoustic *frame-level features* (often referred to as *low-level descriptors, LLDs*) are extracted. Then, the frame-level features are summarised over each instance (sample) using either *functionals* or a *BoAW* representation. Finally, the instance level feature vector is decoded using a *support vector machine* classifier. In comparison to the approach followed by [St10b], no segmentation of the samples to detect the sustain part is done prior to feature extraction.

3.1 Acoustic frame-level features

The frame-level features capture the short-term characteristics of an audio signal within a short interval (‘frame’) in time, where the signal is supposed to be quasi-stationary. In other words, frame-level features are numeric descriptors of certain properties of the audio signal. Those properties can be the *energy* of the waveform or its frequency of changing its sign (*zero crossing rate*), but usually, also much more complex descriptors are employed. These are, first of all, *spectral* features, describing the amplitude of the signal in certain frequency bands, but also the predominant frequency of the signal (*pitch, F0*) and the ratio

[5] The samples denoted by *noFX* and the samples denoted by *EQ* in the database constitute our *noFX* class.

of *harmonic* content in the signal, as opposed to noise. Over the past few years, there has been a lot of research on the suitability of feature sets for different audio recognition tasks. In the field of *computational paralinguistics*, dealing with the recognition of speakers' states and traits (such as emotion and health) from speech, one driving factor has been the series of *ComParE* challenges at the *Interspeech* conferences, beginning in 2009 [SSB09]. In the 2013 challenge, the large-scale ComParE feature set [Sc13] has been introduced, which is based on 65 frame-level features and their corresponding *delta* coefficients, i. e., their differences between adjacent frames, in total, a feature vector of size 130 per frame. Many features used in computational paralinguistics, such as *spectral*, *cepstral* (such as Mel-frequency cepstral coefficients, *MFCC*), *energy-related*, or *pitch-related* features, have also proven their suitability for acoustic scene classification [GSR13] and MIR [We13], which motivates their investigation in the context of audio effect recognition.

Feature extraction is done with the toolkit openSMILE [Ey13] on overlapping frames with a hop size (frame shift) of 10 ms. Two different frame sizes were used, depending on the feature type: For *energy*, *spectral*, and *cepstral* features, a size of 20 ms was employed; for *zero-crossing rate* and all features related to *pitch* (e. g., *voicing probability* and *jitter*), a size of 60 ms was employed. A detailed list of all frame-level features is given in Table 2; further details on their computation are given in [Ey15].

3.2 Functionals

As the short-term information of the frame-level features is usually not meaningful for the classification of a whole audio sample (instance), they need to be summarised over the whole sample. A straightforward approach to this is using so-called *functionals*, i. e., statistics for each of the 65 frame-level features (coefficients) and their delta coefficients (deltas) over each sample. Those statistical measures can, e. g., be *arithmetic mean*, *standard deviation*, *skewness*, *kurtosis*, *quartiles*, *inter-quartile ranges*, *percentiles*, *regression coefficients and error*, *total ranges of coefficients*, *peak values*, *rise time*, and *linear prediction gain and coefficients*.

A set of meaningful functionals is already defined in the ComParE feature set [Sc13]. Some of the functionals are not or only applied to the delta coefficients, which results in a total number of 6 373 features per instance. Detailed information on the functionals are given in [Ey15]. An overview of the feature types investigated in this contribution is given in Table 2.

3.3 Bag-of-audio-words

BoAW are summarising the frame-level features of one instance in a histogram, i. e., in a representation counting the term frequencies of each frame-level feature within an audio sample. The order of the input features within the sample is not taken into account in this approach. As opposed to linguistic words, which are discrete, the frame-level features are vectors of continuously valued numbers, so, there would be an infinite amount of audio

Feature type	**# Frame-level features**	**# Instance level features (coefficients)**	**# Instance level features (deltas)**	**# Instance level features (coef. + deltas)**
RMS energy	1	54	46	100
Zero-crossing rate	1	54	46	100
Spectral features	15	810	690	1 500
Auditory spectrum	28	1 512	1 288	2 800
MFCCs 1-14	14	756	644	1 400
F0	1	44	39	83
logHNR	1	39	39	78
Voicing probability	1	39	39	78
Jitter (local + DDP)	2	78	78	156
Shimmer (local)	1	39	39	78
Total	**65**	**3 425**	**2 948**	**6 373**

Tab. 2: Feature types present in the ComParE feature set. MFCC: Mel-frequency cepstral coefficients; HNR: Harmonics-to-noise ratio

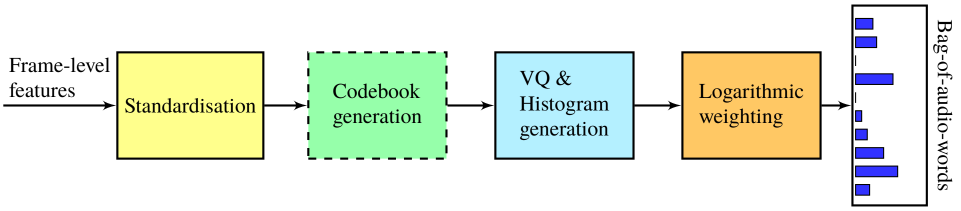

Fig. 1: Bag-of-audio-words processing chain.

words theoretically. This is why in BoAW (and in BoVW), the input is subject to *vector quantisation*. In this step, the Euclidean distance between a given feature vector and all 'audio words' in a given *codebook*, a fixed-size list of meaningful frame-level feature vectors, is computed.

In Figure 1, the main steps of the BoAW computation are shown. As a first step, all frame-level features are *standardised* in an *on-line* approach. This preprocessing step is important if features of different ranges are combined in one BoAW, as features of a high dynamic range would have a larger impact on the BoAW computation without standardisation. The codebooks are generated simply by a *random sampling* of the frame-level features in the respective training set. This method has proven to compete with *k-means clustering* in a BoAW framework [Ra13]. However, we employed the initialisation step of *k-means++ clustering* [AV07], which results in a more 'balanced' random selection of audio words in the input feature space. As a final step, the logarithm is taken from the term frequencies, to reduce their dynamic range. For the whole processing chain displayed in Figure 1, we use our open-source crossmodal bag-of-words toolkit openXBOW [SS16].

3.4 Classifier

The functionals or BoAW representation of the frame-level features is finally fed into a *support vector machine* (SVM) classifier, originally introduced by Cortes and Vapnik [CV95] in 1995. This method is based on the principle of finding an optimum hyperplane in a multidimensional feature space that separates all data samples into the given classes. This hyperplane is optimised in a way that it has a maximum distance to the samples closest to the hyperplane ('support vectors'). The feature space of the samples can implicitly be transformed to a higher dimensional space using *kernels*, however, given the high dimensionality of the employed feature vectors, this step is usually not providing any benefit. Thus, the so-called *linear kernel* is used, where a fast implementation exists with the toolkit LIBLINEAR [Fa08]. We use the *1-vs-all* multi-class scheme with an *L2-regularized L2-loss support vector classification (primal)* solver and a *bias* of 1.

One important parameter to be optimised in the context of SVM is the *complexity*. With this parameter, a certain amount of 'error' can be allowed during optimisation of the hyperplane, which effects that the orientation of the hyperplane is not influenced too much by outliers in the training data and thus prevents overfitting if selected carefully. We optimise the complexity parameter in the range $[10^{-6}, 10^{-5}, ..., 1]$ using 2-fold cross-validation on the training partition, as described in the following section.

Standardisation of features, i. e., the linear transformation of each single feature to zero mean and unit standard deviation over the whole data set, is a common step prior to classification. This step usually helps both to increase the accuracy of the classifier and to reduce the computational complexity of its optimisation. Standardisation can be done either in an *off-line* or in an *on-line* manner. In the first case, the parameters (mean and standard deviation) for each feature are estimated from each data partition (training set or test set) independently; in the latter case, the parameters are estimated from the training set only. Due to the large range of the numbers in the functionals of COMPARE, standardisation (either on-line or off-line) is always applied here, while it is not necessary for BoAW.

4 Experiments and results

With the proposed approaches, we ran our evaluations on the dataset introduced in Section 2. All experiments were conducted directly as a classification problem of all 11 classes. This is aligned with the "experiment 4" in the work by Stein et al. [St10b]. The experiment splits up into 4 different parts, where each part is an evaluation conducted on another subset of the IDMT-SMT-AUDIO-EFFECTS dataset. They are summarised in Table 3. On each part of the dataset, experiments can be conducted with two permutations, where one subset is used for training (Train) and the other for evaluation (Eval). For this reason, the experiments are abbreviated by a *letter*, followed by an *index* for the permutation.

In order to prevent *overfitting* and to obtain reliable results, we optimise the complexity of the SVM on a reasonable split of the respective training set. For the experiments A and B, the split is made based on the instrument configuration mentioned in the dataset description,

Name	**Acronym** [St10b]	**Train**	**Eval**
A1	BS-MO	Yamaha BB604	Warwick Corvette
A2		Warwick Corvette	Yamaha BB604
B1	GIT-MO	Schecter Diamond	Chester Strat.
B2		Chester Strat.	Schecter Diamond
C1	BS-GIT	Yamaha BB604 + Warwick Corvette	Schecter Diamond + Chester Strat.
C2		Schecter Diamond + Chester Strat.	Yamaha BB604 + Warwick Corvette
D1	GIT-MP	Schecter + Chester (monophonic)	Schecter + Chester (polyphonic)
D2		Schecter + Chester (polyphonic)	Schecter + Chester (monophonic)

Tab. 3: Overview over the experiments.

for the experiments C and D, each part of the split consist of all samples of one instrument. Results are reported only on the respective evaluation (Eval) partition. This is aligned with the evaluation using "contextual information" by Stein et al. [St10b]. The performance is given in terms of the *unweighted average recall (UAR)*, which is equal to the *accuracy*, as the 11 classes are balanced in all subsets.

Generally, the experiments C and D are the most challenging, because different effect settings were used for bass and guitar, and effects may have a completely different impact on monophonic and polyphonic sounds, especially, effects adding harmonics, such as *overdrive* and *distortion*. Furthermore, the impact of the missing segmentation of the sample might be larger in the polyphonic case.

Feature type	**A1**	**A2**	**B1**	**B2**	**C1**	**C2**	**D1**	**D2**
ALL	72.5	**77.0**	**89.3**	94.8	**61.5**	**56.5**	9.1	**13.6**
ALL, no deltas	67.5	70.9	86.3	90.5	56.0	52.4	**9.4**	12.2
ALL, only deltas	**73.5**	74.5	87.9	**95.3**	60.3	55.7	9.1	12.9

Tab. 4: UAR [%] based on ComParE + Functionals, **on-line** standardisation.

Table 4 shows the performance of the effect recognition using the functionals-based representation of the ComParE feature set, with *on-line* standardisation. Results are shown for the fusion of all feature types, with an additional report of the UARs for a feature set using either not or only the delta coefficients. While the accuracies for the experiments A-C are satisfactory, the results for experiment D (monophonic vs polyphonic) are very poor and almost on chance level. We supposed, that the reason for this is the employed on-line standardisation, which is not able to tackle systematic differences in the range of features between training and evaluation set. E. g., it is obvious that polyphonic samples will have a higher mean energy than monophonic samples. Off-line standardisation makes indeed sense for functionals-based feature sets as the features are directly related to feature values while it is not relevant for BoAW features, as the magnitude of the 'bag' is not depending on the magnitude of the frame-level features. However, off-line standardisation always requires a large ‘in-domain’ set (in our case the evaluation set), where the parameters (mean and standard deviation) are tuned on. So, it is not really robust, as it always requires the knowledge of the domain (e. g., polyphonic guitar recordings) a data sample has been taken from.

Corresponding results with off-line standardisation are shown in Table 5. It is obvious that the results for all experiments are clearly better when using off-line standardisation. Detailed accuracies for all 10 feature types are provided. For lack of space, we do not give detailed results on the performance of coefficients and deltas for each feature type. However, the qualitative overall (ALL) finding that the delta coefficients are more meaningful for experiments C and D goes also for the single feature types.

Feature type	**A1**	**A2**	**B1**	**B2**	**C1**	**C2**	**D1**	**D2**
RMS energy	64.2	62.8	76.1	77.0	57.4	44.4	25.5	36.5
Zero-crossing rate	30.5	30.8	34.5	38.9	29.8	24.3	1.6	11.3
Spectral features	75.7	73.4	92.0	93.3	**73.1**	54.9	40.9	45.4
Auditory spectrum	71.2	66.2	85.6	87.7	57.3	56.1	66.0	36.5
MFCCs 1-14	52.5	50.8	65.8	67.7	37.2	40.0	31.2	25.4
F0	32.4	33.8	55.1	55.1	33.0	27.8	15.3	21.9
logHNR	52.1	51.0	63.1	62.7	42.3	28.3	13.7	11.3
Voicing probability	43.4	43.0	63.1	63.7	42.7	34.6	18.1	30.2
Jitter (local + DDP)	35.4	32.9	57.8	54.2	30.2	20.1	12.3	15.8
Shimmer (local)	46.7	43.1	63.3	62.3	36.7	31.9	15.9	16.0
ALL	**83.3**	**82.0**	**96.7**	**97.8**	70.5	**63.7**	59.8	50.7
ALL, no deltas	79.4	78.0	95.1	96.7	68.7	58.4	56.4	46.0
ALL, only deltas	79.5	78.3	94.9	96.3	71.9	63.2	**63.5**	**51.1**

Tab. 5: UAR [%] based on ComParE + Functionals, **off-line** standardisation.

Next, the functionals are compared to BoAW representations. The *codebook size* is a key parameter during the tuning process. We realised that the results get better when the codebook size is increased, also if the size of the resulting BoAW vectors is much larger than that of the default ComParE feature set with functionals. Making a fair comparison of features and BoAW is a challenging task. On the one hand, audio words are randomly selected, and so the need of a larger word space is well-founded; on the other hand, classification in a larger feature space is usually easier, and also the feature space of functionals could be easily augmented, with the risk of overfitting. However, BoAW can be scaled easily using larger codebook sizes, whereas further functionals must be manually defined. Firstly, in Table 6, the classification results with BoAW are presented for experiment B. The mean value over the two permutations is shown for all feature types and a fusion of all features. For the fusion of all feature types, the bags derived for each feature type are fused and not the frame-level features. This early fusion approach is provided directly by openXBOW. The codebook size for each feature type is either the same as the dimensionality of functionals (x1), twice (x2), or four times (x4) this size.

Feature types	**BoAW**			**BoAW, no deltas**			**BoAW, only deltas**		
	CS factor			CS factor			CS factor		
	x1	x2	x4	x1	x2	x4	x1	x2	x4
RMS energy	59.5	62.1	62.9	34.8	34.2	33.5	50.9	51.6	51.6
Zero-crossing rate	33.1	32.8	33.5	27.7	27.7	27.7	26.5	27.2	27.1
Spectral features	58.2	59.0	61.6	41.8	41.7	43.7	61.9	63.6	67.9
Auditory spectrum	53.7	56.9	58.1	42.4	45.2	48.1	51.8	54.3	56.7
MFCCs 1-14	48.5	50.7	51.8	34.3	34.4	35.1	50.3	52.2	53.8
F0	23.3	26.9	29.9	21.0	24.5	25.8	19.4	34.4	38.1
logHNR	50.8	56.9	60.5	36.3	37.4	37.2	45.2	52.7	53.6
Voicing probability	43.4	48.5	50.9	30.3	29.9	29.0	39.2	46.1	48.0
Jitter (local + DDP)	28.7	28.6	33.0	34.4	37.3	39.3	12.9	21.5	29.4
Shimmer (local)	45.2	48.7	50.5	34.9	35.2	35.1	37.6	40.2	40.3
ALL	83.8	86.2	87.9	70.5	72.0	75.6	85.9	90.5	**90.8**

Tab. 6: UAR [%] based on ComParE + BoAW, detailed results for experiment B (average of B1 & B2). Codebooks sizes (CS) are always a multiple (with given factor) of the dimensionality of functionals displayed in Table 2.

We provide the results for all 4 experiments with an optimised setting in Table 7. As before, the frame-level features are split into their feature types prior to BoAW generation and then fused into one feature vector for SVM training. Results with a codebook size of 2 000 for each feature type are given, which provides even better results. On average, BoAW outperform functionals with on-line standardisation, but not with off-line standardisation.

Feature type	**A1**	**A2**	**B1**	**B2**	**C1**	**C2**	**D1**	**D2**
ALL	72.0	69.4	87.2	88.0	64.9	55.1	41.1	45.7
ALL, no deltas	64.0	58.9	73.6	76.8	49.4	41.8	22.4	40.1
ALL, only deltas	**76.5**	**72.4**	**90.9**	**92.8**	**67.7**	**58.2**	**35.8**	**62.1**

Tab. 7: UAR [%] based on ComParE + BoAW with a codebook size of 2 000 for each feature type.

Finally, we present the class-specific recall for each feature type in Table 8. To demonstrate this, the average performance of experiments B1 and B2 for the functionals-based approach with off-line standardisation was selected, as this has proven the best overall performance.

5 Discussion

In our best proposed approach, the full ComParE feature set with functionals and off-line standardisation, the results partially compete with those reported by Stein et al. [St10b], shown in Table 9. While the results for the guitar-only experiments (B) are slightly surpassed, they are slightly lower for experiment A, but meaningfully lower for experiments C and D. The reason for this comparably worse performance might be their cutting of the *attack* parts, which are known to be influenced by the instrument type and sound to a large extent, while this is less the case for the *sustain* part.

Feature type	Effect types										
	Chorus	Distortion	FeedbackDelay	Flanger	NoFX	Overdrive	Phaser	Reverb	SlapbackDelay	Tremolo	Vibrato
RMS energy	50.3	97.8	75.9	72.1	84.7	95.6	46.4	83.5	71.6	98.0	66.2
ZCR	10.0	96.6	15.0	29.8	32.7	80.2	19.9	24.9	11.5	58.5	24.4
Spectral feat.	80.7	99.3	98.0	89.2	94.1	95.2	79.3	98.8	96.0	99.4	88.9
Auditory sp.	78.4	100	86.9	86.8	90.3	93.9	63.3	92.4	83.8	98.3	79.5
MFCCs 1-14	54.9	98.6	75.6	75.2	55.7	81.8	74.3	67.1	45.1	49.3	57.1
F0	48.9	63.0	57.3	59.9	51.0	62.8	33.9	55.9	48.3	32.8	92.4
logHNR	53.3	85.3	75.6	57.5	62.1	79.9	38.0	73.8	52.4	64.4	49.5
Voicing prob.	40.5	90.8	86.5	33.2	58.9	67.4	43.2	86.4	88.9	45.8	55.8
Jitter	30.1	63.8	82.1	62.9	26.0	53.9	36.3	91.7	55.6	37.9	75.7
Shimmer	35.8	83.9	64.9	55.7	61.1	55.9	36.4	90.4	51.1	92.8	62.5
ALL	95.6	100	98.9	96.9	94.8	97.1	91.8	99.6	96.5	100	98.8

Tab. 8: UAR [%] per effect class based on ComParE + BoAW, mean over both permutations of experiment B (guitar) is shown.

Experiment	Acronym, see Table 3	UAR [%]
A1 & A2	BS-MO	84.5
B1 & B2	GIT-MO	95.7
C1 & C2	BS-GIT	76.0
D1 & D2	GIT-MP	63.3

Tab. 9: Reference results by [St10b] with optimum feature set in terms of UAR [%]. The average over both permutations 1 & 2 is shown.

Overall, the *zero-crossing rate* is the least useful feature in most cases, whereas the *spectral features* provide best results throughout the experiments. However, the optimum performance is usually achieved when using the fusion of all features. Interestingly, better results are obtained in some cases, when considering only the delta coefficients of the features. This applies to the experiments C and D when using functionals with off-line standardisation and always when using BoAW. Based on our findings, the only advantage of BoAW is that they can be used in an on-line manner, i. e., standardisation of the frame-level features can be done using the parameters derived during training, while there is no need for standardisation of the final 'bags'. Also off-line standardisation of the frame-level features was tried in this context, but the results were slightly lower.

It seems to be the general case that the deltas are more meaningful than the actual coefficients, which indicates that the audio effects manifest in the short-term evolution of the signal rather than in its static properties. This motivates also the investigation, if the deltas of the delta coefficients (*double deltas* or *acceleration* coefficients) imply meaningful information on the present audio effects. Furthermore, a *hierarchical* approach as investigated by Stein et al. [St10b] might be worth further consideration.

Table 8 shows that recognition of the effects *Distortion* and *Tremolo* works best in experiment B, while the detection of *Phaser* is the worst. These findings are also valid for the other experiments and for the BoAW approach. It might be surprising that even the discrimination between even closely related effects, such as *Chorus* and *Flanger*, works with a low level of confusion.

Finally, two major limitations are addressed, which could raise troubles when applying the proposed recognition system to real music recordings. Firstly, the dataset has been generated with only a single digital audio workstation. In real music recordings, however, the employed effects processors will be different ones which usually do not employ the same algorithm or circuit for processing. Secondly, detection of the *delay* effects worked well in our experiments. However, in a dataset of only single notes, this task is relatively easy, as the delay effects are the only ones resulting in two distinct attacks. In a real-life database of continuous recordings, it would be much more difficult to discriminate between a delay effect and a repeated note.

6 Conclusion and outlook

We have seen that, given the high accuracy for the experiments with only guitar, the problem of recognition of effects in a single-instrument track can almost be considered solved. We have given some insights into the usefulness of certain acoustic feature types for the task and seen that the delta coefficients are often more meaningful than the actual coefficients. The bag-of-audio-words approach is a meaningful alternative to using functionals, especially when the standardisation cannot be done off-line. A reasonable fusion of classifiers specialised on specific audio effects groups will further improve the accuracy. The polyphonic recognition might be tackled by using corresponding training data.

Future work must further investigate how well a recogniser performs on data generated by different digital audio workstations or effects processors. From the methodological side, *multi-task learning* seems suitable for the task at hand [Zh16]. As in the field of speech recognition, where the knowledge of age and gender is useful for the training process, it will be beneficial, to have access to side information, in order to learn intrinsic dependencies within the data. This side information is – in the case of the database at hand – the played note, the played string, and the plucking style.

Acknowledgements

The research leading to these results has received funding from the European Union’s Seventh Framework and Horizon 2020 Programmes through Grant Agreements No. 338164 (ERC StG iHEARu) and No. 645094 (IA SEWA).

References

[Ab17] Abeßer, J.; Frieler, K.; Cano, E.; Pfleiderer, M.; Zaddach, W.-G.: Score-informed analysis of tuning, intonation, pitch modulation, and dynamics in jazz solos. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 25(1):168–177, 2017.

[AV07] Arthur, D.; Vassilvitskii, S.: k-means++: The advantages of careful seeding. In: Proceedings of the 18th annual ACM-SIAM symposium on Discrete Algorithms. SIAM, New Orleans, USA, pp. 1027–1035, 2007.

[Be13] Benetos, E.; Dixon, S.; H., D. Giannoulis; Kirchhoff; Klapuri, A.: Automatic music transcription: challenges and future directions. Journal of Intelligent Information Systems, 41(3):407–434, 2013.

[CV95] Cortes, C.; Vapnik, V.: Support-vector networks. Machine learning, 20(3):273–297, 1995.

[Di12] Dittmar, Christian; Cano, Estefanía; Abeßer, Jakob; Grollmisch, Sascha: Music information retrieval meets music education. In: Proceedings of Dagstuhl Follow-Ups. volume 3, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Germany, 2012.

[Ey13] Eyben, F.; Weninger, F.; Groß, F.; Schuller, B.: Recent developments in openSMILE, the Munich open-source multimedia feature extractor. In: Proceedings of the 21st ACM International Conference on Multimedia (ACM MM). Barcelona, Spain, pp. 835–838, 2013.

[Ey15] Eyben, F.: Real-time speech and music classification by large audio feature space extraction. Springer International Publishing, 2015.

[Fa08] Fan, R.-E.; Chang, K.-W.; Hsieh, C.-J.; Wang, X.-R.; Lin, C.-J.: LIBLINEAR: A library for large linear classification. The Journal of Machine Learning Research, 9:1871–1874, 2008.

[GPF15] Grzeszick, R.; Plinge, A.; Fink, G. A.: Temporal acoustic words for online acoustic event detection. In: Proceedings of the 37th German Conference on Pattern Recognition (GCPR). Springer, Aachen, Germany, pp. 142–153, 2015.

[GSR13] Geiger, J. T.; Schuller, B.; Rigoll, G.: Large-scale audio feature extraction and SVM for acoustic scene classification. In: Proceedings of the 2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics. IEEE, New Paltz, USA, pp. 1–4, 2013.

[Ke14] Kehling, C.; Abeßer, J.; Dittmar, C.; Schuller, G.: Automatic tablature transcription of electric guitar recordings by estimation of score-and instrument-related parameters. In: Proceedings of DAFx. Erlangen, Germany, pp. 219–226, 2014.

[KV09] Klapuri, A.; Virtanen, T.: Automatic music transcription. Springer, 2009.

[PA12] Pancoast, S.; Akbacak, M.: Bag-of-audio-words approach for multimedia event classification. In: Proceedings of INTERSPEECH. ISCA, Portland, USA, pp. 2105–2108, 2012.

[Po15] Pokorny, F.; Graf, F.; Pernkopf, F.; Schuller, B.: Detection of negative emotions in speech signals using bags-of-audio-words. In: Proceedings of the 1st International Workshop on Automatic Sentiment Analysis in the Wild (WASA 2015) held in conjunction with ACII 2015. AAAC, IEEE, Xi'an, China, pp. 879–884, 2015.

[Ra13] Rawat, S.; Schulam, P. F.; Burger, S.; Ding, D.; Wang, Y.; Metze, F.: Robust audio-codebooks for large-scale event detection in consumer videos. In: Proceedings of INTERSPEECH. ISCA, Lyon, France, pp. 2929–2933, 2013.

[RHG08] Riley, M.; Heinen, E.; Ghosh, J.: A text retrieval approach to content-based audio hashing. In: Proceedings of ISMIR. Philadelphia, USA, pp. 295–300, 2008.

[Sc13] Schuller, B.; Steidl, S.; Batliner, A.; Vinciarelli, A.; Scherer, K.; Ringeval, F.; Chetouani, M.; Weninger, F.; Eyben, F.; Marchi, E.; Mortillaro, M.; Salamin, H.; Polychroniou, A.; Valente, F.; Kim, S.: The INTERSPEECH 2013 computational paralinguistics challenge: social signals, conflict, emotion, autism. In: Proceedings of INTERSPEECH. ISCA, Lyon, France, pp. 148–152, 2013.

[Sc16] Schmitt, M.; Janott, C.; Pandit, V.; Qian, K.; Heiser, C.; Hemmert, W.; Schuller, B.: A bag-of-audio-words approach for snore sounds' excitation localisation. In: Proceedings of ITG Speech Communication. Paderborn, Germany, pp. 230–234, 2016.

[SDP12] Song, Y.; Dixon, S.; Pearce, M.: A survey of music recommendation systems and future perspectives. In: Proceedings of the 9th International Symposium on Computer Music Modeling and Retrieval. London, U. K., 2012.

[SRS16] Schmitt, M.; Ringeval, F.; Schuller, B.: At the border of acoustics and linguistics: bag-of-audio-words for the recognition of emotions in speech. In: Proceedings of INTERSPEECH. ISCA, San Francisco, USA, pp. 495–499, 2016.

[SS16] Schmitt, M.; Schuller, B. W.: openXBOW – Introducing the Passau open-source crossmodal bag-of-words toolkit. preprint arXiv:1605.06778, 2016.

[SSB09] Schuller, B.; Steidl, S.; Batliner, A.: The Interspeech 2009 emotion challenge. In: Proceedings of INTERSPEECH. ISCA, Brighton, U. K., pp. 312–315, 2009.

[St10a] Stein, M.: Automatic detection of multiple, cascaded audio effects in guitar recordings. In: Proceedings of the 13th International Conference on Digital Audio Effects (DAFx). Graz, Austria, pp. 4–7, 2010.

[St10b] Stein, M.; Abeßer, J.; Dittmar, C.; Schuller, G.: Automatic detection of audio effects in guitar and bass recordings. In: Proceedings of the AES 128th Convention. Audio Engineering Society, San Francisco, USA, 2010.

[We13] Weninger, F.; Eyben, F.; Schuller, B. W.; Mortillaro, M.; Scherer, K. R.: On the acoustics of emotion in audio: what speech, music and sound have in common. Frontiers in Psychology, section Emotion Science, Special Issue on Expression of emotion in music and vocal communication, 4(Article ID 292):1–12, 2013.

[YSY13] Yeh, C.-C. M.; Su, L.; Yang, Y.-H.: Dual-layer bag-of-frames model for music genre classification. In: Proceedings of ICASSP. IEEE, Vancouver, Canada, pp. 246–250, 2013.

[Zh16] Zhang, Y.; Weninger, F.; Ren, Z.; Schuller, B.: Sincerity and deception in speech: two sides of the same coin? A transfer- and multi-task learning perspective. In: Proceedings of INTERSPEECH. ISCA, San Francisco, USA, pp. 2041–2045, 2016.

[Zö11] Zölzer, U.: DAFX: Digital Audio Effects. John Wiley & Sons, 2011.