CHORD AND BEAT MULTI-TASK LEARNING WITH DCNNS

First Author
Affiliation1
author1@ismir.edu

Second Author
Retain these fake authors in
submission to preserve the formatting

Third Author
Affiliation3
author3@ismir.edu

ABSTRACT

In this work we evaluate a multi-task learning approach to beat tracking and chord detection. Both tasks are relevant and well known in the MIR community. Multi-task learning aims at training one model simultaneously on two tasks which share common properties or depend on common features in the training data. The goal is to improve performance by leveraging additional context provided by this shared information. As a side effect, a wider range of datasets becomes available for training. Additionally, multi-task learning yields a single model instead of one model for each task. As network architecture we use a dilated convolutional neural network. We train and evaluate models on a wide range of diverse datasets for chord detection and beat tracking in single- as well as multi-task learning scenarios. The performance of the resulting systems are benchmarked against comparable state-of-the-art approaches in both tasks. We find that the beat tracking models for both single and multi-task training perform on par to the state-of-the-art. For chord detection we can observe performance improvements when looking at multi-tasks results compared to single-task models while being comparable to state-of-the-art results.

1. INTRODUCTION

In this work, we address two core music information research (MIR) tasks: beat tracking and chord detection.

Beat tracking is the task of identifying the fundamental rhythmical grid that defines the tempo of a musical piece. Usually, a stream of virtual quarter or eighth notes represents the beat, e.g. clicks of a metronome. That means, the beat is a more or less regular pulse of events over time, aligned to the tempo grid of the music. Beat information can be helpful or even necessary for many higher level MIR tasks such as tempo estimation [1], transcription [2], or segmentation [3].

In chord detection, the task is to identify the played chords within a musical piece. In this context, the chords in a piece can be defined as the resulting perceived impression of chords, e.g. the chords that could be used in a solo accompaniment arrangement (e.g. solo guitar chords). While chords have a large vocabulary and can be built modularly, in the context of this work, we focus only on identifying the base note (12 semitones) and the two main variants of chords: major and minor. Adding a *no-chord class* for cases where no major or minor chord is being played, this results in 25 classes. This is a common simplification (e.g. [4]) of the task, which is also evaluated in the MIREX Chord estimation task [1] .

The two tasks rely on common information (e.g. note onset events) but are somewhat complementary: While beat tracking mainly focuses on the temporal and rhythmical structure of the piece, chord detection analyzes the harmonic structure within the music. However, chord changes most often happen in context to the beat, i.e. exactly on the beat or down-beat (start of new measures) or in a well defined place between two beats. For those reasons, like for many tasks in MIR (cf. [2], [5]), combining these two tasks in a multi-task machine learning system appears beneficial.

Generally, the idea of multi-task learning is to combine two or more tasks which share common properties, and train them together in order to leverage the additional context [6]. Furthermore, by combining two tasks, usually a larger volume of data can be used for training. This is because different tasks usually come with non-intersecting datasets, and by using a multi-task approach all datasets with only partial (in our case either chord or beat) annotations can be used. The expected benefits are: *i.* faster convergence during training, *ii.* smaller models (compared to two single-task systems), *iii.* better generalization, and, therefore, *iv.* improved performance.

In this work, we present a beat and chord multi-task system based on a fully convolutional artificial neural network. As input for the network, constant-Q-like spectrograms are fed into a 2D convolutional stack. Convolutional neural networks (CNNs) have been shown to be well suited for analyzing musical content from a spectrogram representation [2, 7–11]. Additionally the network features a stack of dilated convolutions. This architecture is chosen as it allows to create a wide receptive field (in the order of tens of seconds) which is required especially in the context of beat tracking. Using dilated convolutions to achieve a large receptive field has been shown to be effective for generating and analyzing time series data, especially audio signals [1, 12, 13]. Prior to this, large receptive fields were often realized by using recurrent neural networks (RNNs), particularly LSTMs [8, 14]. The dis-

[1] https://www.music-ir.org/mirex/wiki/2020:Audio_Chord_Estimation

advantage of LSTMs is that they can only be trained in a sequential manner, which—depending on the hardware—usually results in much longer training times when dealing with large amounts of data.

2. RELATED WORK

Both beat tracking and chord detection are tasks with a long history. There exists a variety of beat tracking approaches based on classical signal processing methods as well as machine learning, and the same is true for chord detection. Since an exhaustive discussion of past approaches in both tasks is beyond the scope of this work, we focus on the most recent and relevant work wrt. our approach.

In terms of beat tracking, these are the works by Böck et al. and Davies et al. [1, 5, 13–15].

The systems presented by Böck et al. in 2014 and 2016 [14, 15], use the same long short-term memory (LSTM [16]) neural network models, combined with a dynamic Bayesian network (DBN), and are trained in similar fashion. They mainly differ in how the network ensembles are built and in the addition of down-beat tracking in 2016. These works set a performance baseline for beat tracking which is still valid today, seven years later.

In 2019, Böck and Davies introduce a dilated convolutional network (DCNN) for beat tracking [13]. They use this approach and further build upon it, adding tempo multi-tasking [1] as well as down-beat tracking [5]. We extend upon the DCNN model architectures introduced in these works, adding the capability for chord detection.

Integrating tempo prediction with a beat tracking system was previously explored by Krebs et al. [17] in 2013. They introduce a model which infers tempo, bar position, and rhythm patterns using a dynamic Bayesian network (DBN). Holzapfel et al. [18] introduce a similar method for metrical analysis of non-western music. A further improvement of these methods was introduced by Krebs et al. [19] in 2016.

Using multi-task learning in the audio domain, especially combining another MIR task, other than tempo, with beat tracking has also been investigated before. In 2017, Vogl et al. [2] evaluate the potential of using beat, down-beat, and drum detection multi-tasking. They compare CNNs, RNNs, and CRNNs in this context and investigate if prior knowledge of beat information is beneficial for drum transcription and find that this benefit remains when using multi-task learning.

The most relevant works in terms of chord transcription for this paper are by Korzeniowski and Widmer [4, 20]. In [4] they introduce a fully convolutional VGG-style architecture for chord detection, using a conditional random field (CRF) as post-processing step. They use the activations of the penultimate layer of the CNN as feature embeddings for the CRF. While they use the same 25 chord classes vocabulary, they use 10fps spectrograms in contrast to 100fps in our work. In [20], they focus on the potential of language models as a means of post-processing the chord change sequence. Since in this work, we are mainly interested in the effect of the multi-task approach on the machine learning model, we do not further investigate the effects of post-processing.

Humphrey and Bello [21] identify the limited amount of training examples for each of the many chord classes as a problem for machine learning. In a later work [22], they evaluate GMM and CNN based chord recognition systems specifically designed for a large vocabulary chord detection task. They also provide an analysis of the subjective nature of chord annotations and discuss evaluation methods. To tackle the problem of sparsely populated chord classes, Korzeniowski and Widmer [4] utilize data augmentation in the form of pitch shifting, to artificially increase the number of chord examples for each root note. The idea of using transformations of the spectrogram as means for data augmentation in the context of audio processing was also discussed by Schlüter and Grill in a work on singing voice detection [23].

While the local audio information can be sufficient to identify chords, it has been shown that a receptive field on the order of seconds or tens of seconds is beneficial also for chord detection. This can be explained by the fact that chord progressions and key, which are global properties, are strong hints for local chord detection. Leveraging this context, Boulanger-Lewandowski et al. [24] present a chord detection system based on RNNs.

McFee and Bello [25] combine convolutional and recurrent layers in a convolutional-recurrent neural network (CRNN) as part of a encoder/decoder approach to tackle a large vocabulary (170 classes) chord detection task. In order to be able to handle the large number of classes, they simplify the chords by discarding inversions, suppressed, and additional notes, and then representing chords by their root note (12) and 14 qualities.

Another recent approach to chord transcription is presented by Chen and Su [26]. They draw ideas from natural language processing using a transformer network architecture to train a multi-task model for key, harmonic functions, and chord recognition. This work demonstrates the capabilities of transformer models in the context of audio analysis, as well as the potential of multi-task learning for harmony related tasks. The presented chord recognition performance in this work constitutes a new state-of-the art, however because the performance was only evaluated on the *McGill Billboard* dataset, which was not available to use, a direct comparison is not possible. Since Chen and Su [26] provide a comparison with the approach of Korzeniowski and Widmer [27] on the *McGill Billboard* dataset, it is reasonable to assume that the transformer model proposed in [26] would outperform the approach presented in this paper. However, given the higher complexity of the model and combination of multiple task in the context of harmony in [26], we are confident that our work contributes relevant and new information since we isolate the effects of combining beat tracking with chord detection using a compact, simple, fast to train, and versatile convolutional network.

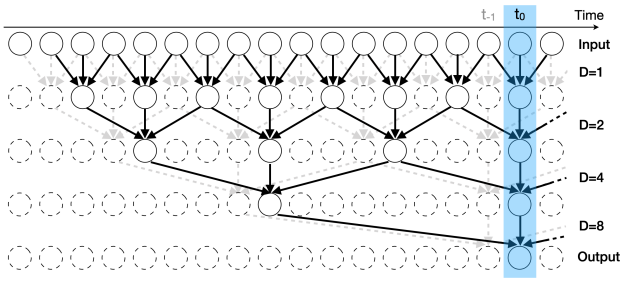

Figure 1. Dilated convolutional stack. Non-causal system (dashed connections from future), kernel size 3. Grey dashed connections represent processing for step t_{-1}.

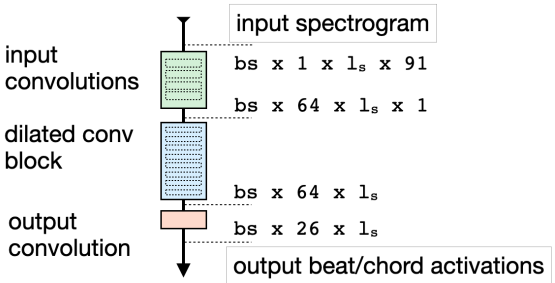

Figure 2. Building block overview of neural network architecture. Numbers to the left represent tensor shapes, bs denotes batch size, l_s sequence length.

3. METHOD

The choice of using a DCNN architecture was made for several reasons: *i.* it has been shown that DCNNs can successfully be used to analyze time series features [5, 12, 13], *ii.* as an alternative to RNNs they are faster to train as they can leverage the parallel processing powers of modern GPUs and TPUs better, and *iii.* they easily allow to model different receptive fields. Figure 1 shows how dilated convolutions are used to construct a wide receptive field, using exponentially increasing dilation factors (D). In principle, it is possible to build this architecture as a causal (real-time capable) or non-causal model. In our case, to be comparable to the state-of-the-art approaches in the literature, we chose the non-causal approach.

The overall processing pipeline for a neural network based audio analysis system usually consists of the following blocks: *i.* audio preprocessing/feature extraction, *ii.* neural network, and *iii.* network output post processing. These processing steps of the presented approach are discussed in detail in the following Sections 3.1, 3.2, and 3.3.

3.1 Signal Preprocessing

The goal of the signal preprocessing step is to extract a suitable signal representation from the audio files for further processing in the network. We first normalize different audio formats to mono, 44.1k Hz sampling rate. After that, a short-time Fourier transform (STFT) using 4096-sample Hann windows and a 441-sample hop-size is performed. The real values of the resulting complex spectrogram are then filtered using logarithmically spaced triangular frequency filters referring to semitones within the equal temperament tuning scale, thus 12 bins per frequency doubling are used (12 semitones per octave). By using a frequency range from 30Hz to 17kHz, a total number of 91 frequency bins are obtained. Finally, the log values of the resulting log-frequency spectrogram are used as input features for the neural network. The resulting spectrogram has a frame rate of 100fps, which is suitable for beat tracking, but ten times the usual frame rate for chord detection [4, 14].

3.2 Network Architecture

The used architecture consists of the following three building blocks: The input block is a VGG-style [28] block consisting of 2D convolution layers with 3x3 kernels, and 2D max pooling layers with 1x3 kernels ($t \times f$). The task of this block is to process the spectrogram frames and identify relevant structures and features in the spectrogram. Ideally, the 3x3 filters are able to find the harmonic structures relevant for chord detection, as well as locate power spikes which correspond to onsets relevant for beat tracking.

The second block contains a temporal convolutional network consisting of 1D dilated convolution layers with a kernel size of 5. The task of this block is to provide the receptive field in order to be able to identify long term structures, relevant for beat tracking and chord changes. This is achieved by stacking 8 layers with dilation rates $D = 2^n, n \in [0..7]$, to obtain a receptive field of 1027 spectrogram frames (including the receptive field of the input block), which translates to 10.27 seconds. Note, that by using a higher input feature frame rate, for chord detection this comparable wide receptive field is adequate [2].

Finally, the third block is a 1D convolution layer with kernel size 1, which acts as a dense layer and provides 26 output channels. The first 25 channels are used for a softmax output representing the 25 chord classes (12 root notes, with major/minor variants each, and one *no-chord* class) and one sigmoid output for beat activation. As loss we use cross-entropy for the chord class softmax outputs and binary cross-entropy for the sigmoid beat activation output. Padding keeps the sequence length l_s constant throughout the network which enables processing of arbitrary length sequences. Table 1 contains all details for layer parameterization.

To improve generalization and stabilize training, following regularization techniques are used: Between convolutional layers and activation functions, batch-normalization [29] is applied. Additionally dropout [30] with $p = 0.1$ is applied after each convolutional layer. The used network is a fusion of the architectures used in [13] and [4] and uses maximums for parameters from both: e.g. number/channels of input convolutions.

[2] C.f. 105 frames receptive field at 10fps frame rate, which translates to 10.5 seconds [4].

type	channels	kernel	pad	output
Input	1	–	–	l_s x 91
Conv2D ELU	1/32	3x3	0x2	l_s x 89
MaxPool2D	32/32	1x3	0x0	l_s x 29
Conv2D ELU	32/32	3x3	0x2	l_s x 27
MaxPool2D	32/32	1x3	0x0	l_s x 9
Conv2D ELU	32/64	3x3	0x2	l_s x 7
MaxPool2D	64/64	1x3	0x0	l_s x 2
Conv2D ELU	64/64	1x2	0x0	l_s x 1
type	**channels**	**kernel**	**pad**	**dilation**
Conv1D ELU	64/64	5	2	1
Conv1D ELU	64/64	5	4	2
Conv1D ELU	64/64	5	8	4
Conv1D ELU	64/64	5	16	8
Conv1D ELU	64/64	5	32	16
Conv1D ELU	64/64	5	64	32
Conv1D ELU	64/64	5	128	64
Conv1D ELU	64/64	5	256	128
Conv1D ELU	64/25	1	0	1

Table 1. Layer setup for network. Every conv layer uses batch-norm and dropout ($p = 0.1$). l_s, *sequence length* number of input feature frames.

3.3 Postprocessing

Both network outputs for beat tracking, as well as chord detection have to be post-processed to yield beat positions and chord labels, respectively. In case of beat outputs with sigmoid activation function, potential beat positions are expected to be indicated by spikes towards the value of 1. In the simplest case, one could use a peak picking approach to detect those peaks. However, as shown in [15], using a more refined postprocessing method yields better performance. For simplicity, we use the same DBN postprocessing method introduced in [15] and used in following works [1, 5, 13, 14]. The DBN method is available in the madmom [31] library and consists of individual state machines for different tempi in the expected range for music. Tempo variations are possible due to low probability state transitions between different tempo state chains. For more details we refer to the original work in [15].

For chord detection, simply selecting the class with maximum probability from the softmax outputs of the network yields usable chord labels. We apply simple median filtering to mitigate short spurious chord changes. In order to be able isolate effects of the multi-task learning model we opted to avoid intricate post-processing, also given that related works use rather complex methods to achieve performance gains <2%, e.g. CRF [4], RNN [20].

3.4 Network Training

The network is trained using a 8-fold cross-validation setup. The 8 splits are drawn across all used datasets according to the split definitions provided by [14] and [4]. For one training run, one split is reserved for testing, another split is used for validation after each epoch, and the remaining are used for updating the parameters during training. The data is used in chunks of 1030 frame sequences obtained using a hop-size of 250 samples. For optimization we use Adam [32] in combination with a mini batch strategy using batches of 256 randomly drawn sequences and an initial learning rate $l_r = 0.02$. Early stopping is used, i.e. the training is halted as soon as the validation loss stops decreasing during 5 epochs (patience). Additionally two refinement runs are performed after early stopping. For refinement, the best performing model on the validation set is used and further trained with a reduced learning rate: $l_r = l_r * 0.2$.

Used Datasets	*# files*	*length*	*chord*	*beat*
Beatles [33, 34]	180	8h 09m	✓	✓
R. Williams [35]	65	4h 31m	✓	✓
Ballroom [17, 36]	685	5h 57m		✓
Rock [37]	200	12h 53m		✓
Hainsworth [38]	222	3h 19m		✓
RWC pop [39]	100	6h 47m	✓	✓
Queen [34]	19	1h 13m	✓	
Zweieck [34]	18	58m	✓	
SMC* [40]	217	2h 25m		✓
GTZAN* [41, 42]	999	8h 20m		✓

Table 2. Overview of used datasets. Datasets marked with an asterisk (*) are held out for testing only.

4. DATASETS AND EVALUATION

For training and testing, we use diverse datasets (see Table 2) in a 8-fold cross-validation setup. We use datasets which provide both beat and chord annotations as well as datasets that only have either beat or chord annotations. The chord annotations are represented by timestamps and chord symbols following the chord notation used in MIREX [43]. The provided chord symbols are reduced to only major and minor chords using the *mir_eval* [44] Python library. Annotations for beat tracking consist of timestamps and optional downbeat or beat number indications, which are ignored. As training targets for the sigmoid beat output a synthetic beat activation function is created which is zero for all frames, except for frames with beat annotations, where it is one. To increase the number of examples per chord class, we use the same data augmentation approach as in [4] by pitch shifting the signal in the spectrogram by ± 4 semitones randomly for 20% of the examples in a batch.

All datasets, except SMC and GTZAN are used for training, validation, and testing of the neural network. The two excluded sets are held out for testing generalization capabilities. This is according to common practice in the literature, due to some challenging properties of the datasets in the context of beat tracking [14, 15].

For evaluation metrics for beat tracking we use a subset of [33], which are commonly used [10, 13–15, 19].

Chord Results	WCSR	Segmentation
Isophonics (Beatles, Queen, and Zweieck)		
CNN [4]	*0.8290*	-
single-task	0.7917	0.8619
multi-task	0.8072	0.8659
Robbie Williams		
CNN [4]	*0.8280*	-
single-task	0.8064	0.8750
multi-task	0.8203	0.8792
RWC Pop		
CNN [4]	*0.8250*	-
single-task	0.7938	0.8678
multi-task	. 0.8068	0.8714

Table 3. Chord detection performance results.

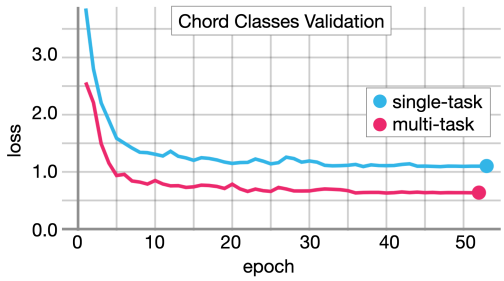

Figure 3. Chord detection validation losses for a single split, single task v.s. multi-task. Loss for multi-task decreases faster and further than for single-task.

For chord detection we use Weighted Chord Symbol Recall (WCSR) and segmentation, both used in the literature [4, 22, 25, 26].

5. RESULTS

5.1 Experiments

We run three different experiments using the network architecture from section 3.2, training on: *i.* only-beat datasets, using beat targets and beat loss for updates (*single-task beat*), *ii.* only-chord datasets, using chord targets and chord loss for updates (*single-task chord*), and *iii.* datasets with both beat and chord annotations, combining both losses for updates (*multi-task*). In case of the multi-task training, more precisely, we sum up beat and chord losses while weighting the beat loss with a factor of 15. This is due to the imbalances of chord classes (26) and beat outputs (1) as well as of chord and beat annotations per unit of time. The exact weighting factor was determined by experimentation using the validation loss curves (beat and chord loss curves should have similar shapes).

Once the three models are trained, we evaluate them using the test sets for each respective cross-validation split and the hold out test data for beat tracking. Table 3 shows the chord detection results for both the single-task chord model and multi-task model. Table 4 shows the beat tracking results for both the single-task beat model and multi-task model.

5.2 Discussion

The main value of Table 4 is in showing that all compared beat models roughly perform the same, although the oldest one is over 7 years old. However, we can show that by using similarly sized models compared to solo beat tracking models in the literature, the network still has the capacity to learn a second task without a decrease in performance. This hints towards a glass ceiling for beat tracking on these datasets that might have been reached. Further performance improvements are probably due to overfitting, and real improvement might only be achievable by fixing annotation errors, or introducing models that can focus on handling fringe cases, like odd meters or extreme tempi. This finding is no surprise, therefore we focus on the chord results for more insights.

In Table 3, the chord prediction performance consistently improves by >1% when using chord and beat multi-tasking. In addition to the fact that the used network architecture is a compromise for beat tracking and chord recognition and has to split capacities for two tasks, it is worth noting that a performance increase can still be observed. Furthermore, also the input features and frame rate are a compromise for chord detection. Korzienowski et al. [4, 20, 27] consistently use a lower frame rate of 10fps, and higher frequency resolution spectrograms using 24 bins per octave and a 8192 samples STFT window.

While the multi-task models' performance is still below the state-of-the-art, the baseline approach uses a CRF post-processing model which usually increases the performance by 1-2% compared to raw outputs [27]. Thus we deem the results of the multi task model comparable to the state-of-the art.

Furthermore, we can observe small but consistent improvements in terms of segmentation. That implies that by using a multi-task model alongside beat tracking, position of chord changes in time has improved.

In Figure 3, additionally we can observe that for the multi-task training, the chord validation loss decreases faster and lies below the solo chord loss.

6. CONCLUSION

We investigated a multi-task approach to beat tracking and chord detection using a DCNN architecture trained on both beat tracking and chord detection datasets. For beat tracking, we achieve results on par with state-of-the-art approaches for both single- and multi-task training. For chord detection we achieve comparable results to the state-of-the-art discounting CRF post-processing, in the case of multi-task training. We can observe a consistent increase of performance when comparing single-task and multi-task chord detection models. In light of the simplicity of the model, which is trained on two tasks, and several compromises made for chord detection, we deem the results relevant and a basis for further developments into the direction of multi-task learning for music information research tasks.

Beat Results	F1	CMLc	CMLt	AMLc	AMLt	D
Ballroom						
TCN [13]	0.933	0.864	0.881	0.909	0.929	3.456
BLSTM [15]	0.917	0.832	0.849	0.905	0.926	3.539
BLSTM [14]	0.938	0.872	0.892	0.932	0.953	3.397
single-task	0.928	0.860	0.880	0.904	0.925	3.425
multi-task	0.914	0.824	0.843	0.900	0.920	2.641
Hainsworth						
TCN [13]	0.874	0.755	0.795	0.882	0.930	3.518
BLSTM [15]	0.884	0.769	0.808	0.873	0.916	3.507
BLSTM [14]	0.871	0.732	0.784	0.849	0.910	3.395
single-task	0.883	0.773	0.820	0.865	0.921	3.487
multi-task	0.874	0.762	0.804	0.866	0.917	2.679
Beatles						
TCN [13]	-	-	-	-	-	-
BLSTM [15]	0.880	-	-	-	-	-
BLSTM [14]	0.918	-	-	-	-	-
single-task	0.915	0.763	0.822	0.837	0.904	2.938
multi-task	0.920	0.771	0.836	0.844	0.915	2.366
Robbie Williams						
TCN [13]	-	-	-	-	-	-
BLSTM [15]	-	-	-	-	-	-
BLSTM [14]	-	-	-	-	-	-
single-task	0.913	0.773	0.813	0.855	0.901	3.224
multi-task	0.909	0.776	0.806	0.861	0.901	2.460
Rock						
TCN [13]	-	-	-	-	-	-
BLSTM [15]	-	-	-	-	-	-
BLSTM [14]	-	-	-	-	-	-
single-task	0.899	0.729	0.784	0.825	0.898	3.576
multi-task	0.909	0.767	0.818	0.849	0.921	2.744
RWC						
TCN [13]	-	-	-	-	-	-
BLSTM [15]	0.877	-	-	-	-	-
BLSTM [14]	0.943	-	-	-	-	-
single-task	0.910	0.792	0.832	0.873	0.920	2.683
multi-task	0.910	0.778	0.814	0.884	0.930	2.694
SMC*						
TCN [13]	0.543	0.315	0.432	0.462	0.632	1.574
BLSTM [15]	0.529	0.296	0.428	0.383	0.567	1.460
BLSTM [14]	0.516	0.307	0.406	0.429	0.575	1.514
single-task	0.534	0.299	0.418	0.423	0.605	1.492
multi-task	0.521	0.302	0.428	0.423	0.598	0.994
GTZAN*						
TCN [13]	0.843	0.695	0.715	0.889	0.914	3.096
BLSTM [15]	0.864	0.750	0.768	0.901	0.927	3.071
BLSTM [14]	0.843	0.695	0.715	0.889	0.914	3.096
single-task	0.873	0.762	0.788	0.888	0.924	3.083
multi-task	0.868	0.750	0.777	0.885	0.921	2.383

Table 4. Beat tracking performance results and baselines. Performance are high and close over models and datasets. Datasets marked with an asterisk (*) were used exclusively for testing, not during cross-validation.

7. REFERENCES

[1] S. Böck, M. E. P. Davies, and P. Knees, "Multi-task learning of tempo and beat: Learning one to improve the other." in *20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019, pp. 486–493.

[2] R. Vogl, M. Dorfer, G. Widmer, and P. Knees, "Drum transcription via joint beat and drum modeling using convolutional recurrent neural networks," in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017, pp. 150–157.

[3] S. Lattner, M. Grachten, K. Agres, and C. E. C. Chacón, "Probabilistic segmentation of musical sequences using restricted boltzmann machines," in *Proceedings of the 5th Mathematics and Computation in Music Conference (MCM)*, 2015, pp. 323–334.

[4] F. Korzeniowski and G. Widmer, "A fully convolutional deep auditory model for musical chord recognition," in *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2016, pp. 1–6.

[5] S. Böck and M. E. P. Davies, "Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation," in *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[6] R. Caruana, "Multitask learning," in *Learning to learn*. Springer, 1998, pp. 95–133.

[7] J. Schlüter and B. Lehner, "Zero-Mean Convolutions for Level-Invariant Singing Voice Detection," in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[8] R. Vogl, G. Widmer, and P. Knees, "Towards multi-instrument drum transcription," in *Proceedings of the 21st International Conference on Digital Audio Effects (DAFx)*, 2018, pp. 57–64.

[9] C. Southall, R. Stables, and J. Hockman, "Automatic drum transcription for polyphonic recordings using soft attention mechanisms and convolutional neural networks," in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017, pp. 606–612.

[10] S. Durand, J. P. Bello, B. David, and G. Richard, "Feature adapted convolutional neural networks for downbeat tracking," in *Proceedings of the 41st IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 296–300.

[11] J. Schlüter and S. Böck, "Improved musical onset detection with convolutional neural networks," in *Proceedings of the 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 6979–6983.

[12] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[13] M. E. P. Davies and S. Böck, "Temporal convolutional networks for musical audio beat tracking," in *2019 27th European Signal Processing Conference (EUSIPCO)*, 2019, pp. 1–5.

[14] S. Böck, F. Krebs, and G. Widmer, "Joint beat and downbeat tracking with recurrent neural networks," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016.

[15] ——, "A multi-model approach to beat tracking considering heterogeneous music styles." in *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014, pp. 603–608.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] F. Krebs, S. Böck, and G. Widmer, "Rhythmic pattern modeling for beat and downbeat tracking in musical audio," in *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2013, pp. 227–232.

[18] A. Holzapfel, F. Krebs, and A. Srinivasamurthy, "Tracking the "odd": Meter inference in a culturally diverse music corpus," in *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014, pp. 425–430.

[19] F. Krebs, S. Böck, M. Dorfer, and G. Widmer, "Downbeat tracking using beat-synchronous features and recurrent neural networks," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016, pp. 129–135.

[20] F. Korzeniowski and G. Widmer, "Improved chord recognition by combining duration and harmonic language models," in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, 2018, pp. 10–17.

[21] E. J. Humphrey and J. P. Bello, "Rethinking automatic chord recognition with convolutional neural networks," in *Proceedings of the 11th International Conference on Machine Learning and Applications*. IEEE, 2012, pp. 357–362.

[22] ——, "Four timely insights on automatic chord estimation." in *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, 2015, pp. 673–679.

[23] J. Schlüter and T. Grill, "Exploring data augmentation for improved singing voice detection with neural networks," in *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, 2015.

[24] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Audio chord recognition with recurrent neural networks." in *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*, 2013, pp. 335–340.

[25] B. McFee and J. P. Bello, "Structured training for large-vocabulary chord recognition." in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017, pp. 188–194.

[26] T.-P. Chen and L. Su, "Harmony transformer: Incorporating chord segmentation into harmony recognition," in *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019, pp. 259–267.

[27] F. Korzeniowski and G. Widmer, "On the futility of learning complex frame-level language models for chord recognition," in *Audio Engineering Society Conference: 2017 AES International Conference on Semantic Audio*. Audio Engineering Society, 2017.

[28] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. [Online]. Available: http://arxiv.org/abs/1409.1556

[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," http://arxiv.org/abs/1502.03167, 2015. [Online]. Available: http://arxiv.org/abs/1502.03167

[30] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[31] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer, "madmom: a new python audio and music signal processing library," https://arxiv.org/abs/1605.07008, 2016.

[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. [Online]. Available: https://arxiv.org/abs/1412.6980v9

[33] M. E. P. Davies, N. Degara, and M. D. Plumbley, "Evaluation methods for musical audio beat tracking algorithms," Centre for Digital Music, Queen Mary University of London, Tech. Rep. Technical Report C4DM-TR-09-06, 2009.

[34] C4DM Queen Mary University of London, "Isophonics.net," http://isophonics.net/datasets.

[35] B. DiGiorgi, M. Zanoni, A. Sarti, and S. Tubaro, "Automatic chord recognition based on the probabilistic modeling of diatonic modal harmony," in *8th International Workshop on Multidimensional Systems (nDS))*, 2013, pp. 145–150.

[36] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano, "An experimental comparison of audio tempo induction algorithms," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1832–1844, 2006.

[37] T. De Clercq and D. Temperley, "A corpus analysis of rock harmony," *Popular Music*, pp. 47–70, 2011.

[38] S. W. Hainsworth and M. D. Macleod, "Particle filtering applied to musical tempo tracking," *EURASIP Journal on Advances in Signal Processing*, vol. 2004, no. 15, pp. 1–11, 2004.

[39] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "Rwc music database: Popular, classical and jazz music databases." in *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, vol. 2, 2002, pp. 287–288.

[40] A. Holzapfel, M. E. P. Davies, J. R. Zapata, J. L. Oliveira, and F. Gouyon, "Selective sampling for beat tracking evaluation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 9, pp. 2539–2548, 2012.

[41] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on speech and audio processing*, vol. 10, no. 5, pp. 293–302, 2002.

[42] U. Marchand and G. Peeters, "Swing ratio estimation," in *Proceedings of the 18th International Conference on Digital Audio Effects (DAFx)*, 2015.

[43] C. Harte, M. B. Sandler, S. A. Abdallah, and E. Gómez, "Symbolic representation of musical chords: A proposed syntax for text annotations." in *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, 2005, pp. 66–71.

[44] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis, "mir_eval: A transparent implementation of common mir metrics," in *Proceedings of the 15th International Conference on Music Information Retrieval (ISMIR)*, 2014.