AUDIO CHORD RECOGNITION WITH A HYBRID RECURRENT NEURAL NETWORK

Siddharth Sigtia* **Nicolas Boulanger-Lewandowski**† **Simon Dixon***
$*$ Centre for Digital Music, Queen Mary University of London, London, UK
† Dept. IRO, Université de Montréal, Montréal (QC), H3C 3J7, Canada
$*${s.s.sigtia,s.e.dixon}@qmul.ac.uk

ABSTRACT

In this paper, we present a novel architecture for audio chord estimation using a hybrid recurrent neural network. The architecture replaces hidden Markov models (HMMs) with recurrent neural network (RNN) based language models for modelling temporal dependencies between chords. We demonstrate the ability of feed forward deep neural networks (DNNs) to learn discriminative features directly from a time-frequency representation of the acoustic signal, eliminating the need for a complex feature extraction stage. For the hybrid RNN architecture, inference over the output variables of interest is performed using beam search. In addition to the hybrid model, we propose a modification to beam search using a hash table which yields improved results while reducing memory requirements by an order of magnitude, thus making the proposed model suitable for real-time applications. We evaluate our model's performance on a dataset with publicly available annotations and demonstrate that the performance is comparable to existing state of the art approaches for chord recognition.

1. INTRODUCTION

The ideas presented in this paper are motivated by the recent progress in end-to-end machine learning and neural networks. In the last decade, it has been shown that given a large dataset, deep neural networks (DNNs) are capable of learning useful features for discriminative tasks. This has led complex feature extraction methods to be replaced with neural nets that act directly on raw data or low level features. Current state-of-the-art methods in speech recognition and computer vision employ DNNs for feature extraction [12]. In addition to feature learning, recurrent neural networks (RNNs) have been shown to be very powerful models for temporal sequences [9, 12]. In the field of Music Information Retrieval (MIR), various studies have applied neural network based models to different tasks [3, 11, 15]. These experiments have been motivated by the fact that hand-crafting features to extract musically relevant information from audio is a difficult task. Existing approaches in MIR would benefit greatly if feature extraction could be automated.

Audio chord recognition is a fundamental problem in MIR (see [13] for a review). At a high level, popular chord recognition algorithms follow a pipeline similar to the one followed in speech. Most systems are comprised of an *acoustic model* which is used to process the acoustic information present in the audio signal. The estimates of the acoustic model are further refined by a *language model* that models the temporal relationships and structure present in sequences of chord symbols. Our proposed approach deviates from existing approaches in two fundamental ways. We use DNNs to learn discriminative features from a time-frequency representation of the audio. This is contrary to the common approach of extracting chroma features (and their many variants) as a preprocessing step. Secondly, we generalise the popular method of using a Hidden Markov Model (HMM) language model with a more powerful RNN based language model. Finally, we combine the acoustic and language models using a hybrid RNN architecture previously used for phoneme recognition and music transcription [5, 14].

In the past, RNNs have been applied to chord recognition and music transcription in a sequence transduction framework [3, 4]. However, these models suffer from an issue known as *teacher forcing*, which occurs due to the discrepancy between the training objective and the way the RNN is used at test time. During training, RNNs are trained to predict the output at any time step, given the correct outputs at all preceding steps. This is in contrast to how they are used at test time, where the RNN is fed predictions from previous time steps as inputs to the model. This can lead to an unsuitable weighting of the acoustic and symbolic information, which can quickly cause errors to accumulate at test time. The hybrid RNN architecture resolves this issue by offering a principled way for explicitly combining acoustic and symbolic predictions [14].

The hybrid RNN model outputs a sequence of conditional probability distributions over the output variables (Section 3). The structure of the graphical model makes the problem of exactly estimating the most likely sequence of outputs intractable. Beam search is a popular heuris-

[†] NB is currently working at Google Inc., Mountain View, USA

tic graph search algorithm which is used to decode conditional distributions of this form. Beam search when used for decoding temporal sequences is fundamentally limited by the fact that sequences that are quasi-identical (differ at only few time steps) can occupy most of the positions within the beam, thus narrowing the range of possibilities explored by the search algorithm. We propose a modification to the beam search algorithm which we call *hashed beam search* in order to encourage *diversity* in the explored solutions and reduce computational cost.

The rest of the paper is organised as follows: Section 2 describes the feature learning pipeline. Section 3 briefly introduces the hybrid RNN architecture. Section 4 describes the proposed modification to the beam search algorithm. Experimental details are provided in Section 5, results are outlined in Section 6 and the paper is concluded in Section 7.

2. FEATURE LEARNING

We follow a pipeline similar to the one adopted in [3, 15] for feature extraction. We transform the raw audio signal into a time-frequency representation with the constant-Q transform [6]. We first down-sample the audio to 11.025 kHz and compute the CQT with a hop-size of 1024 samples. The CQT is computed over 7 octaves with 24 bins per octave yielding a 168 dimensional vector of real values. One of the advantages of using the CQT is that the representation is low dimensional and linear in pitch. Computing the short-time Fourier transform over long analysis windows would lead to a much higher dimensional representation. Lower dimensional representations are useful when using DNNs since we can train models with fewer parameters, which makes the parameter estimation problem easier.

After extracting CQT frames for each track, we use a DNN to classify each frame to its corresponding chord label. As mentioned earlier, DNNs have been shown to be very powerful classifiers. DNNs learn complex non-linear transformations of the input data through their hidden layers. In our experiments we used DNNs with 3 hidden layers. We constrained all the layers to have the same number of hidden units to simplify the task of searching for good DNN architectures. The DNNs have a softmax output layer and the model parameters are obtained using maximum likelihood estimation.

Once the DNNs are trained, we use the activations of the final hidden layer of the DNN as features. In our experiments we observed that the acoustic model performance was improved ($\sim 3\%$ absolute improvement in frame-level accuracy) if we provided each frame of features with context information. Context information was provided by performing mean and variance pooling over a context window around the central frame of interest [3]. A context window of length $2k + 1$ is comprised of the central frame of interest, along with k frames before and after the central frame. In our experiments we found that a context window of 7 frames provided the best results.

We trained the network with mini-batch stochastic gradient descent. Instead of using learning rate update schedules, we use ADADELTA which adapts the learning rate over iterations [18]. In our experiments we found Dropout was essential to improve generalisation [16]. We found a Dropout rate of 0.3 applied to all layers of the DNN to be optimal for controlling overfitting. Once the models are trained, we use the model that performs best on the validation set to extract features. In our experiments, the best performing model had 100 hidden units in each layer. Figure 1 is a graphical representation of the feature extraction pipeline. In section 6, we compare DNN acoustic models with different feature inputs.

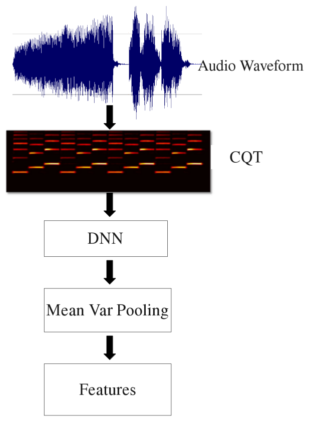

Figure 1. Feature Extraction Pipeline

3. HYBRID RECURRENT NEURAL NETWORKS

Similar to language, chord sequences are highly correlated in time. We propose exploiting this structure for audio chord estimation using hybrid RNNs. The hybrid RNN is a generative graphical model that combines the predictions of an arbitrary frame level classifier with the predictions of an RNN language model. For temporal problems, the predictions of the system can be greatly improved by modelling the relationships between outputs, analogous to language modelling in speech. Typically, HMMs are employed in order to model and exploit this structure. Hybrid RNNs generalise the HMM architecture by using powerful RNN language models.

3.1 RNN Language Model

RNNs can be used to define a probability distribution over a sequence $z = \{z_\tau | 0 \leq \tau \leq T\}$ in the following manner:

$$P(z) = \prod_{t=1}^{T} P(z_t|\mathcal{A}_t) \quad (1)$$

where $\mathcal{A}_t \equiv \{z_\tau | \tau < t\}$ is the sequence history at time t. The above factorisation is achieved by allowing the RNN at time $t - 1$ to predict the outputs at the next time step, yielding the conditional distribution $P(z_t|\mathcal{A}_t)$.

The RNN is able to model temporal relationships via its hidden state which at any time t has recurrent connections to the hidden state at $t-1$. The hidden state is updated according to the following equation:

$$h_t = \sigma(W_{zh} z_{t-1} + W_{hh} h_{t-1} + b_h) \tag{2}$$

where W_{zh} are the weights from the inputs at $t-1$ to the hidden units at t, W_{hh} are the recurrent weights between hidden units at $t-1$ and t and b_h are the hidden biases. The form of the hidden state (Equation 2) implies that the predictions at time t are explicitly conditioned on the entire sequence history $\mathcal{A}_t$. This is contrary to HMMs which are constrained by the Markov property. Therefore, theoretically RNNs can model complex and long-term temporal dependencies between outputs. The parameters of the RNN are estimated by using stochastic gradient based methods. Although theoretically very powerful, RNNs are limited by the effectiveness of the optimisation method [2]. The hidden units described above can be replaced by Long-Short Term Memory (LSTM) units in order to improve the parameter estimation and generalisation capabilities of the RNN (see [10] for a review). In our model we use RNNs with LSTM memory units to model the symbolic structure of chord sequences.

3.2 Hybrid Architecture

Given a sequence of acoustic frames x and a sequence of corresponding chord outputs z, the Hybrid RNN model factorises the joint probability of x and z according to the following equation:

$$\begin{aligned} P(z,x) &= P(z_1 \dots z_T, x_1 \dots x_T) \qquad (3)\\ &= P(z_1)P(x_1|z_1)\prod_{t=2}^{T} P(z_t|\mathcal{A}_t)P(x_t|z_t) \\ &\propto P(z_1)\frac{P(z_1|x_1)}{P(z_1)}\prod_{t=2}^{T} P(z_t|\mathcal{A}_t)\frac{P(z_t|x_t)}{P(z_t)}. \end{aligned}$$

By restricting the acoustic model to operate on an acoustic frame x_t independent of previous inputs and outputs, the distributions $P(z_t|\mathcal{A}_t)$ and $P(z_t|x_t)$ can be independently modelled by an RNN and an arbitrary frame-level classifier, respectively. The form of the joint probability distribution makes maximum likelihood estimation of the model parameters using gradient based optimisers easy. The acoustic and language model terms separate out when optimising the log-likelihood and the model parameters can be trained using gradient based methods according to the following equations where Θ_a, Θ_l are parameters of the acoustic and language models, respectively:

$$\frac{\partial \log P(z,x)}{\partial \Theta_a} = \frac{\partial}{\partial \Theta_a}\sum_{t=1}^{T} \log P(z_t|x_t) \tag{4}$$

$$\frac{\partial \log P(z,x)}{\partial \Theta_l} = \frac{\partial}{\partial \Theta_l}\sum_{t=2}^{T} \log P(z_t|\mathcal{A}_t). \tag{5}$$

Although the hybrid RNN has a similar structure (seperate acoustic and language models) to the sequence transduction model in [9], the hybrid RNN explicitly combines the acoustic and language model distributions. The transduction model in [9], models *unaligned* sequences with an implicit exponential duration.

The property that the acoustic and language models can be trained independently has some useful implications. In MIR, it is easier to obtain chord and note transcriptions from the web as compared to audio data due to copyright issues. We can use the abundance of transcribed data to train powerful language models for various tasks, without the need for annotated, aligned audio data.

4. INFERENCE

The hybrid RNN generalises the HMM graph by conditioning z_t on the entire sequence history $\mathcal{A}_t$, as compared to the HMM graph where z_t is only conditioned on z_{t-1} (Equation 3). This conditioning allows musicological structure learnt by the language model to influence successive predictions. One consequence of the more general graphical structure is that at test time, inference over the output variables at t requires knowledge of all predictions made till time t. At any t, the history $\mathcal{A}_t$ is still uncertain. Making estimates in a greedy chronological manner does not necessary yield good solutions. Good solutions correspond to sequences that maximise the likelihood globally.

Beam search is a standard search algorithm used to decode the outputs of an RNN [5, 9, 14]. Beam search is a breadth-first graph search algorithm which maintains only the top w solutions at any given time. At time t, the algorithm generates candidate solutions and their likelihoods at $t+1$, for all the sub-sequences present in the beam. The candidate solutions are then sorted by log-likelihood and the top w solutions are kept for further search. A beam capacity of 1 is equivalent to greedy search and a beam width of N^T is equivalent to an exhaustive search, where N is the number of output symbols and T is the total number of time steps.

Beam search suffers from a pathological condition when used for decoding sequences. Quasi-identical sequences with high likelihoods can saturate the beam. This limits the range of solutions evaluated by the algorithm. This is especially true when decoding long sequences. The performance of beam search can be improved by pruning solutions that are unlikely. The dynamic programming (DP) based pruned beam search algorithm makes better use of the available beam capacity w [3, 5]. The strategy employed for pruning is that at any time t, the most likely sequence with output symbol $z_t \in C$ is considered and other sequences are discarded, where C is the set of output symbols.

Although the DP beam search algorithm performs well in practice [3,5], pruning based on the last emitted symbol is a strict constraint. In the next section we propose a modification to the beam search algorithm that is more general and allows flexible design to enforce *diversity* in the set of solutions that are explored and to reduce computational cost.

4.1 Hashed Beam Search

As discussed before, beam search can lead to poor estimates of the optimal solution due to saturation of the beam with similar sequences. The efficiency of the search algorithm can be improved by pruning solutions that are sufficiently similar to a sequence with higher likelihood. We propose a more general variant of the pruned beam search algorithm where the metric for similarity of sequences can be chosen according to the given problem. We encode the similarity metric in the form of a *hash function* that determines the similarity of 2 sequences. Given 2 solutions with the same hash value, the solution with the higher likelihood is retained.

The proposed algorithm is more general and flexible since it allows the similarity metric to be chosen based on the particular instance of the decoding problem. We describe the algorithm for decoding chord sequences. We let the hash function be the last n emitted symbols. With this hash function, if there are two candidate solutions with the same sequence of n symbols at the end, then the hash function produces the same key and we retain the solution with the higher likelihood. When $n = 1$, the algorithm is equivalent to the DP beam search algorithm. When $n =$ len(sequence), then the algorithm is equivalent to regular beam search. Therefore, by increasing the value of n, we can linearly relax the constraint used for pruning in the DP-like beam search algorithm.

Another generalisation that can be achieved with the hash table is that for each hash key, we can maintain a list of k solutions using a process called chaining [17]. This is more general than the DP beam search algorithm where only the top solution is kept for each output symbol. Algorithm 1 describes the proposed hashed beam search algorithm, while Algorithm 2 describes the beam objects. The time complexity of Algorithm 1 is $O(NTw \log w)$. Even though the time complexity of the proposed algorithm is the same as regular beam search, the algorithm is able to significantly improve performance by pruning unlikely solutions (see Section 6). In Algorithms 1 and 2, s is a subsequence, l is the log-likelihood of s and f_h is the hash function.

Algorithm 1 Hashed Beam Search

```
Find the most likely sequence $z$ given $x$ with a beam width $w$.
$beam \leftarrow$ new beam object
$beam$.insert($0, \{\}$)
for $t$ = 1 to $T$ do
    $new\_beam \leftarrow$ new beam object
    for $(l, s)$ in $beam$ do
        for $z$ in $C$ do
            $l' = \log P_{lm}(z|s) P_{am}(z|x_t) - \log P(z)$
            $new\_beam$.insert($l + l', \{s, z\}$)
    $beam \leftarrow new\_beam$
return $beam$.pop()
```

Although the description of the proposed algorithm has been within the context of decoding chord sequences, various other measures of similarity can be constructed depending upon the problem. For example, for chord and speech recognition, we can use the last n unaligned symbols as the hash function (results with chords were uninteresting). For problems where the predictions are obtained from an RNN and frame-based similarity measures are insufficient, we can use a vector quantised version of the final hidden state as the key for the hash table entry.

Algorithm 2 Description of beam objects given w, k, f_h

```
Initialise beam object
beam.hashQ = dictionary of priority queues$^*$
beam.queue = indexed priority queue of length $w^{**}$
Insert $l, s$ into beam
key= $f_h(s)$
queue = beam.queue
hashQ = beam.hashQ[key]
fits_in_queue = not queue.full() or $l \geq$queue.min()
fits_in_hashQ = not hashQ.full() or $l \geq$hashQ.min()
if fits_in_queue and fits_in_hashQ then
    hashQ.insert($l, s$)
    if hashQ.overfull() then
        item = hashQ.del_min()
        queue.remove(item)
    queue.insert($l, s$)
    if queue.overfull() then
        item = queue.del_min()
        beam.hashQ[$f_h$(item.$s$)].remove(item)
```

* The dictionary maps hash keys to priority queues of length k which maintain (at most) the top k entries at all times.

** An *indexed* priority queue allows efficient random access and deletion [1].

5. EXPERIMENTS

5.1 Dataset

Unlike other approaches to chord estimation, our proposed approach aims to learn the audio features, the acoustic model and the language model from the training data. Therefore, maximum likelihood training of the acoustic and language models requires sufficient training data, depending on the complexity of the chosen models. Additionally, we require the raw audio for all the examples in the dataset in order to train the acoustic model which operates on CQTs extracted from the audio. In order to satisfy these constraints, we use the dataset used for the MIREX Audio Chord Estimation Challenge. The MIREX data is comprised of two datasets. The first dataset is the collected Beatles, Queen and Zweieck datasets [1] . The second dataset is an abridged version of the Billboard dataset [7].

The Beatles, Queen and Zweieck dataset contains annotations for 217 tracks and the Billboard dataset contains annotations for 740 unique tracks. The corresponding audio for the provided annotations are not publicly available and

[1] http://www.isophonics.net/

we had to acquire the audio independently. We were able to collect all the audio for the Beatles, Queen and Zweieck dataset and 650 out of the 740 unique tracks for the Billboard dataset (see footnote[2] for details), leading to a total of 867 tracks for training and testing[3]. Although we are not able to directly compare results with MIREX evaluations due to the missing tracks, we show that the training data is sufficient for estimating models of sufficient accuracy and the results are comparable to the top performing entries submitted to MIREX 2014. Keeping in mind the limited number of examples in the dataset, all the ground truth chord annotations were mapped to the major/minor chord dictionary which is comprises of 12 major chords, 12 minor chords and one *no chord* class. All results are reported on 4-fold cross-validation experiments on the entire dataset. For training the acoustic and language models, the *training* data was further divided into a training (80%) and validation split (20%).

5.2 Acoustic Model Training

The features obtained from the DNN feature extraction stage (Section 2) are input to an acoustic model which provides a posterior probability over chord labels, $P(z_t|x_t)$ given an input feature vector. Similar to the feature extraction, we use DNNs with a softmax output layer to model the probabilities of output chord classes. We train models with 3 hidden layers with varying number of hidden units. The acoustic models are trained on a frame-wise basis, independently of the language models. We use stochastic mini-batch gradient descent with ADADELTA for estimating the DNN parameters. We use a constant Dropout rate of 0.3 on all the DNN layers to reduce overfitting. Dropout was found to be essential for good generalisation performance, yielding an absolute performance improvement of up to 4% on the test set. We used a mini-batch size of 100 and early stopping for training. Training was stopped if the log-likelihood of the validation set did not increase for 20 iterations over the entire training set. Unlike the feature extraction stage, we do not discard any of the trained models. Instead of using only the best performing model on the validation set, we average the predictions of all the trained models to form an *ensemble of DNNs* [8] as the acoustic model. We found that simply averaging the predictions of the acoustic classifiers led to an absolute improvement of up to 3% on frame classification accuracies.

5.3 Language Model Training

As outlined in Section 3, we use RNNs with LSTM units for language modelling. The training data for the language models is obtained by sampling the ground truth chord transcriptions at the same frame-rate at which CQTs are extracted from the audio waveforms. We use RNNs with 2 layers of hidden recurrent units (100 LSTM units each) and an output softmax layer. Each training sequence was further divided into sub-sequences of length 100. The RNNs were trained with stochastic gradient descent on individual sub-sequences, without any mini-batching. Unlike the acoustic models, we observed that ADADELTA did not perform very well for RNN training. Instead, we used an initial learning rate of 0.001 that was linearly decreased to 0 over 1000 training iterations. We also found that a constant momentum rate of 0.9 helped training converge faster and yielded better results on the test set. We used early stopping and training was stopped if validation log-likelihood did not increase after 20 epochs. We used gradient clipping when the norm of the gradients was greater than 50 to avoid gradient explosion in the early stages of training.

[2] www.eecs.qmul.ac.uk/∼sss31

[3] AUDFPRINT was used to align the audio to the corresponding annotations: http://labrosa.ee.columbia.edu/matlab/audfprint/

	Language Model			
	None		LSTM RNN	
Acoustic Model	OR	WAOR	OR	WAOR
DNN-CQT	57.0%	56.5%	62.8%	62.0%
DNN-DNN Feats	69.8%	69.1%	73.4%	73.0%
DNN-CW DNN Feats	**72.9**%	72.5%	**75.5**%	75.0%

Table 1. 4-fold cross-validation results on the MIREX dataset for the major/minor prediction task. DNN-CQT refers to CQT inputs to a DNN acoustic model. DNN-DNN Feats refers to DNN feature inputs to the DNN acoustic model. DNN-CW DNN Feats refers to DNN features with a context window as input to the acoustic model.

6. RESULTS

In Table 1, we present 4-fold cross validation results on the combined MIREX dataset at the major/minor chord level. The metrics used for evaluation are the overlap ratio (OR) and the weighted average overlap ratio (WAOR) which are commonly used for evaluating chord recognition systems (including MIREX). The test data is sampled every 10ms similar to the MIREX evaluations. The outputs of the hybrid model were decoded with the proposed hashed beam search algorithm. A grid search was performed over the decoding parameters and the presented results correspond to the parameters that were determined to be optimal over the training set.

From Table 1, it is clear that the hybrid model improves performance over the acoustic-only models. The results show that the performance of the acoustic model is greatly improved when the input features to the model are learnt by a DNN as opposed to CQT inputs. The performance of the acoustic model is further improved (3% absolute improvement) when mean and variance pooling is performed over a context window of DNN features. It is interesting to note that the relative improvement in performance is highest for the DNN-CQT and DNN-DNN Feats configurations. This is due to the fact that the hybrid model is derived with the explicit assumption that given a state z_t, the acoustic frame x_t is conditionally independent of all state and acoustic vectors occurring at all other times. Applying a context window to the features violates this independence assumption and therefore the relative improvement is diminished. However, the improved performance of the acoustic model

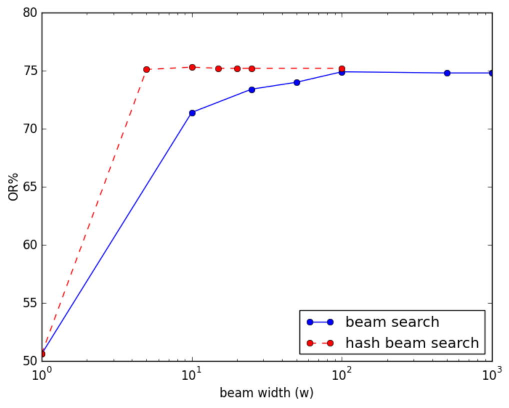

Figure 2. Effect of varying beam width on OR on MIREX data. $n = 2, k = 1$

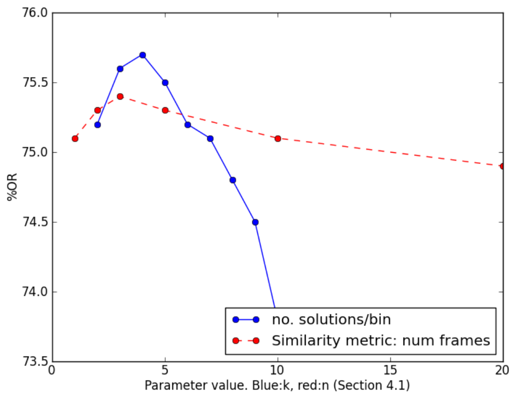

Figure 3. Effect of varying hashed beam search parameters f_h, k on OR on MIREX dataset. $w = 25$.

due to context windowing offsets this loss. Although formal comparisons cannot be made, the accuracies achieved with the hybrid model are similar to the best performing model submitted to the 2014 MIREX evaluation [4] on the Billboard dataset (OR = 75.57%).

To investigate the advantage of the proposed hashed beam search algorithm, we plot the overlap ratio against the beam width. Figure 2 illustrates that the proposed algorithm can achieve marginally better decoding performance at a significant reduction in beam size. As an example, the hashed beam search yields an OR of 75.1% with a beam width of 5, while regular beam search yields 74.7% accuracy with a beam width of 1000. The time taken to run the hash beam search ($w = 5$) over the test set was 5 minutes, as compared to the regular beam algorithm ($w = 1000$) which took 17 hours to decode the test set. The algorithm's ability to yield good performance at significantly smaller beam widths indicates that it performs efficient pruning of similar paths, thus utilising the available beam width more efficiently. The run-times of the algorithm show that it can be used for real-time applications without compromising recognition accuracy.

In addition to the beam width, the hash beam search algorithm allows the user to specify the similarity metric and the number of solutions for each hash table entry. We investigate the effect of these parameters on the OR and plot the results in Figure 3. We let the similarity metric be the previous n frames and observe performance as n is linearly increased for a fixed beam width of 25. From Figure 3 we observe that the performance is quite robust to changes in the number of past frames for small values of n. One possible explanation for the graph is that since the test data is sampled at a frame rate of 10ms, all occurrences of chords last for several frames. Therefore counting the previous n frames, effectively leads to the same metric each time. We experimented with using the previous n *unique* frames as a metric but found that the results deteriorated quite drastically as n was increased. This might reflect the limited memory of RNN language models and the issues caused due to lack of explicit duration modelling. The blue line in Figure 3 illustrates the effect of varying the number of solutions per hash table entry. From this graph we see that performance deteriorates significantly once the number of entries per bin crosses a certain threshold (~ 5). This is due to the fact that maintaining many solutions of the same kind saturates the beam capacity with very similar solutions, limiting the breadth of search.

[4] www.music-ir.org/mirex/wiki/2014:Audio_Chord_Estimation_Results

7. CONCLUSION AND FUTURE WORK

We present a chord estimation system based on a hybrid recurrent neural network and the results are competitive with existing state-of-the-art approaches. We show that DNNs are powerful acoustic models. By learning features, they eliminate the need for complex feature engineering. The hybrid RNN model allows us to superimpose an RNN language model on the acoustic model predictions. Additionally, language models can be trained on chord data from the web without the corresponding audio. The results clearly indicate that the language model helps improve model performance by modelling the temporal relationships between output chord symbols and refining the predictions of the acoustic model. The proposed variant of the beam search algorithm significantly reduces memory usage and run times, making the model suitable for real-time applications.

In the future, we would like to conduct chord recognition experiments on larger datasets. This is because the modelling and generalisation capabilities of neural networks improve with more available data for training. An important issue that remains with respect to RNN language models is the problem of duration modelling. Although RNNs are very good at modelling the transition probabilities between events, durations of each event are not modeled explicitly. For musical applications like chord recognition and music transcription, accurate estimates for durations of note occurrences can further help improve the effectiveness of RNN based language models.

8. REFERENCES

[1] https://pypi.python.org/pypi/pqdict/.

[2] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[3] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Audio Chord Recognition with Recurrent Neural Networks. In *The International Society for Music Information Retrieval Conference (ISMIR)*, pages 335–340, 2013.

[4] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. High-dimensional sequence transduction. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3178–3182. IEEE, 2013.

[5] Nicolas Boulanger-Lewandowski, Jasha Droppo, Mike Seltzer, and Dong Yu. Phone sequence modeling with recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5417–5421. IEEE, 2014.

[6] Judith C Brown. Calculation of a constant Q spectral transform. *The Journal of the Acoustical Society of America*, 89(1):425–434, 1991.

[7] John Ashley Burgoyne, Jonathan Wild, and Ichiro Fujinaga. An Expert Ground Truth Set for Audio Chord Recognition and Music Analysis. In *The International Society for Music Information Retrieval Conference (ISMIR)*, pages 633–638, 2011.

[8] Thomas G Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems*, pages 1–15. Springer, 2000.

[9] Alex Graves. Sequence transduction with recurrent neural networks. In *Representation Learning Workshop, Internation Conference on Machine Learning (ICML)*, 2012.

[10] Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. LSTM: A Search Space Odyssey. *arXiv preprint arXiv:1503.04069*, 2015.

[11] Eric J Humphrey and Juan Pablo Bello. Rethinking automatic chord recognition with convolutional neural networks. In *11th International Conference on Machine Learning and Applications (ICMLA)*, volume 2, pages 357–362. IEEE, 2012.

[12] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[13] Matt McVicar, Raúl Santos-Rodríguez, Yizhao Ni, and Tijl De Bie. Automatic chord estimation from audio: A review of the state of the art. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(2):556–575, 2014.

[14] S. Sigtia, E. Benetos, N. Boulanger-Lewandowski, T. Weyde, Artur S. d'Avila Garcez, and S. Dixon. A Hybrid Recurrent Neural Network for Music Transcription. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2061–2065, Brisbane, Australia, April 2015.

[15] Siddharth Sigtia and Simon Dixon. Improved music feature learning with deep neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6959–6963. IEEE, 2014.

[16] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[17] Clifford Stein, T Cormen, R Rivest, and C Leiserson. *Introduction to algorithms*, volume 3. MIT Press Cambridge, MA, 2001.

[18] Matthew D Zeiler. ADADELTA: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.