Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2007, Article ID 43745, 13 pages
doi:10.1155/2007/43745

Research Article

A Supervised Classification Algorithm for Note Onset Detection

Alexandre Lacoste and Douglas Eck

Department of Computer Science, University of Montreal, Montreal, QC, Canada H3T 1J4

Received 5 December 2005; Revised 9 August 2006; Accepted 26 August 2006

Recommended by Ichiro Fujinaga

This paper presents a novel approach to detecting onsets in music audio files. We use a supervised learning algorithm to classify spectrogram frames extracted from digital audio as being onsets or nononsets. Frames classified as onsets are then treated with a simple peak-picking algorithm based on a moving average. We present two versions of this approach. The first version uses a single neural network classifier. The second version combines the predictions of several networks trained using different hyperparameters. We describe the details of the algorithm and summarize the performance of both variants on several datasets. We also examine our choice of hyperparameters by describing results of cross-validation experiments done on a custom dataset. We conclude that a supervised learning approach to note onset detection performs well and warrants further investigation.

1. INTRODUCTION

This paper is concerned with finding the onset times of notes in music audio. Though conceptually simple, this task is deceivingly difficult to perform automatically with a computer. Consider, for example, the naïve approach of finding amplitude peaks in the raw waveform. This strategy fails except for trivially easy cases such as monophonic percussive instruments. At the same time, onset detection is implicated in a number of important music information retrieval (MIR) tasks, and thus warrants research. Onset detection is useful in the analysis of temporal structure in music such as *tempo identification* and *meter identification*. *Music classification* and *music fingerprinting* are two other relevant areas where onset detection can play a role. In the case of classification, onset locations could be used to significantly reduce the number of frame-level features retained. For example, a sampling method could be used that preferentially selects from frames near-predicted onset locations. A related segmentation strategy for genre classification was used by West and Cox [1]. In the case of music fingerprinting, onset times could be used as the basis of a robust fingerprint vector.

Onset detection is also important in areas involving the structured representation of music. For example, *music editing* (performed using, e.g., a sequencer) can be simplified by using automatic onset detection to segment a waveform into logical parts. Also, onset detection is fundamentally important for the problem of *automatic music transcription*, where a structured symbolic representation (usually a traditional music score) is inferred from a waveform.

Onsets detection algorithms can generally be divided into three steps:

(1) transformation of the waveform to isolate different frequency bands, in general, using either a filter bank or a spectrogram,

(2) enhancement of bands such that note onsets are more salient; this could involve, for example, a filter that detects positive slopes,

(3) peak-picking to select discrete note onsets.

Our main focus is to explore how supervised learning might be used to improve performance within this framework. However, our investigation offers enhancements at each of these three steps. In the first step, we look at different methods for computing and representing the spectrogram as well as at strategies for merging spectrogram frames. In the second step—where we focus most of our attention—we introduce a supervised approach that learns to identify relevant peaks in the output of the first step. Specifically, we train neural networks to provide the best possible onset trace for the peak-picking part. In the third step, we take advantage of a tempo estimate in order to integrate some aspects of rhythmic structure into the peak-picking decision process.

In this paper, we first review the work done in this field with special attention paid to another work done on onset

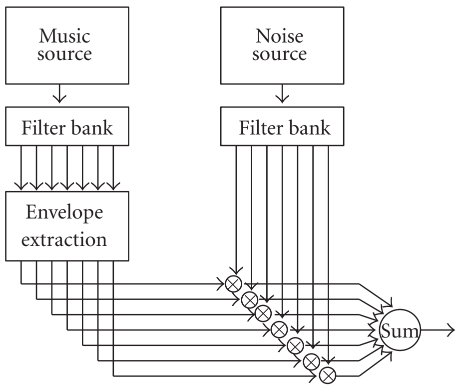

Figure 1: Modulating noise with the energy envelope of different bands from a filter bank retains the rhythmical content of the piece.

detection using machine learning. In Section 3, we describe our algorithm including details about the simpler and more complex variants. In Section 4, we describe a dataset that we built for testing the model. Finally, in Section 5, we present experiment results that report on our investigation of different spectrogram representations and on different network architectures.

2. PREVIOUS WORK

Earlier algorithms developed for onset detection focused mainly on the variation of the signal energy envelope in the time domain. Scheirer [2] demonstrated that much information from the signal can be discarded while still retaining the rhythmical aspect. On a set of test musical pieces, Scheirer filtered out different frequency bands using a filter bank. He extracted the energy envelope for each of those bands, using rectification and smoothing. Finally, with the same filter bank, he modulated a noisy signal with each of those envelopes and merged everything by summation (Figure 1). With this approach, rhythmical information was retained. On the other hand, care must be taken when discarding information. In another experiment, he shows that if the envelopes are summed before modulating the noise, a significant amount of information about rhythmical structure is lost.

Klapuri [3] used the psychoacoustical model developed by Scheirer to develop a robust onset detector. To get better frequency resolution, he employed a filter bank of 21 filters. The author points out that the smallest detectable change in intensity is proportional to the intensity of the signal. Thus $\Delta I/I$ is a constant, where I is the signal's intensity. Therefore, instead of using $(d/dt)A$ where A is the amplitude of the envelope, he used

$$\frac{1}{A}\left(\frac{d}{dt}A\right) = \frac{d}{dt}\log(A). \tag{1}$$

This provides more stable onset peaks and allows lower intensity onsets to be detected. Later, Klapuri et al.used the same kind of preprocessing [4] and won the ISMIR 2004 tempo induction contest [5].

2.1. *Onset detection in phase domain*

In contrast to Scheirer's and Klapuri's works, Duxbury et al. [6–9] took advantage of phase information to track the onset of a note. They found that at steady state, oscillators tend to have predictable phase. This is not the case at onset time, allowing the decrease in predictability to be used as an indication of note onset. To measure this, they collected statistics on the phase acceleration, as estimated by the following equation:

$$\alpha_{k,n} = \operatorname{princarg}\left[\varphi_{k,n} - 2\varphi_{k,(n-1)} + \varphi_{k,(n-2)}\right], \tag{2}$$

where $\varphi_{k,n}$ is the kth frequency bin of the nth time frame from the short-time Fourier transform of the audio signal. The operator princarg maps the angle to the $[-\pi, \pi]$ range. To detect the onset, different statistics were calculated across the range of frequencies including mean, variance, and kurtosis. These provide an onset trace, which can be analyzed by standard peak-picking algorithms. The authors also have combined phase and energy on the complex domain for more robust detection. Results on monophonic and polyphonic music show an increase in performance for phase against energy, and even better performance when combining both.

2.2. *Onset detection using supervised learning*

Only a small amount of work has been done on mixing machine learning and onset detection. In a recent work, Kapanci and Pfeffer [10] used a support vector machine (SVM) on a set of frame features to estimate if there is an onset between two selected frames. Using this function in a hierarchical structure, they are able to find the position of onsets. Their approach mainly focuses on finding onsets in signals with slowly varying change over time such as solo singing.

Davy and Godsill [11] developed an audio segmentation algorithm also using SVM. They classify spectrogram frames into being probable onsets or not. The SVM was used to find a hypersurface delimiting the probable zone from the less probable one. Unfortunately, no clear test was made to outline the performance of the model.

Marolt et al. [12] used a neural network approach for note onset detection. This approach is similar to ours in its use of neural networks, but is otherwise very different. The model used the same kind of preprocessing as by Scheirer in [2], with a filter bank of 22 filters. An integrate-and-fire network was then applied separately to the 22 envelopes. Finally, a multi layer perceptron was applied on the output to accept or reject the onsets. Results were good but the model was only applied to monotimbral piano music.

3. ALGORITHM DESCRIPTION

In this section, we introduce two variants of our algorithm. Both use a neural network to classify frames as being onsets or nononsets. The first variant, SINGLE-NET, follows

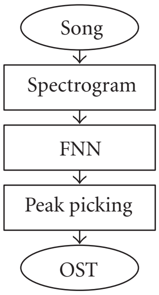

Figure 2: SINGLE-NET flowchart. This simpler variant of our algorithm is comprised of a time-space transform (spectrogram) which is in turn treated with a feed-forward neural network (FNN). The resulting trace is fed into a peak-picking algorithm to find onset times (OSTs).

the process for onset detection described above and shown in Figure 2. Our second variant, MULTI-NET, combines information from (A) *multiple* instantiations of SINGLE-NET, each trained with different hyperparameters and (B) tempo traces gained by running a tempo-detection algorithm on the neural network output vector. The multiple sources of evidence are merged into a feature matrix similar to a spectrogram which is in turn fed back into another feed-forward network, peak picker, and onset detector, see Figure 3.

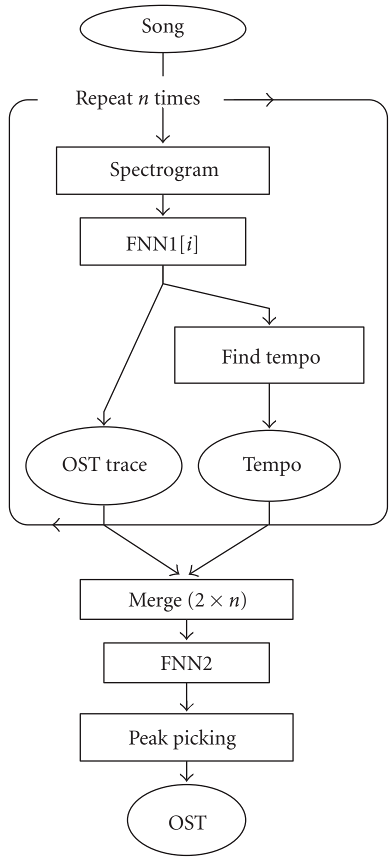

Figure 3: MULTI-NET flowchart. The SINGLE-NET variant is repeated multiple times with different hyperparameters. A tempo-detection algorithm is run on each of the resulting feed-forward neural network (FNN) outputs. The SINGLE-NET outputs and the tempo-detection outputs are then combined using a second neural network.

3.1. Feature extraction

3.1.1. Time-frequency domain transform

Aside from the prediction of global tempo done in the MULTI-NET variant of our algorithm, the information provided to the classification step of the algorithm is local in time. This raises the question of how much local information to integrate in order to achieve best results. Using a parameter search, we concluded that a frame size of at least 50 milliseconds (1/20th of a second) was necessary to generate good results. For a sampling rate of 22050 Hz, this yields $\sim$ 1000 (22050/20) input values per frame for a supervised learning algorithm.

As it is commonly done, we decided to use a time-space transform to lower the dimensionality of the representation and to reveal spectral information in the signal. We focused on the short-time Fourier transform (STFT) and the constant-Q transform [13]. These are discussed separately in the following two sections.

3.1.2. Short-time Fourier transform (STFT)

The short-time Fourier transform is a version of the Fourier transform designed for computing short-time duration frames. A moving window is swept through the signal and the Fourier transform is repeatedly applied to portions of the signal inside the window

$$\text{STFT}(t,\omega) = \int_{-\infty}^{\infty} x(\tau)w^*(\tau - t)e^{-j\omega\tau}d\tau, \qquad (3)$$

where $w(t)$ is the windowing function that isolates the signal for a particular time t and where sequence $x(t)$ is the signal we want to transform, in this case, an audio signal in PCM format.

The discrete version of the STFT is

$$\text{STFT}[n,k] = \sum_{m=-\infty}^{\infty} x[n+m]w[m]e^{-jkm}. \qquad (4)$$

A Hamming window is applied to the signal. By choosing a bigger window width, we get a better frequency resolution but a smaller time resolution. Reducing the window width produces the inverse effect.

3.1.3. Constant-Q transform

The constant-Q transform [13] is similar to the STFT but it has two main differences:

(i) it has a logarithmic frequency scale;
(ii) it has a variable window width.

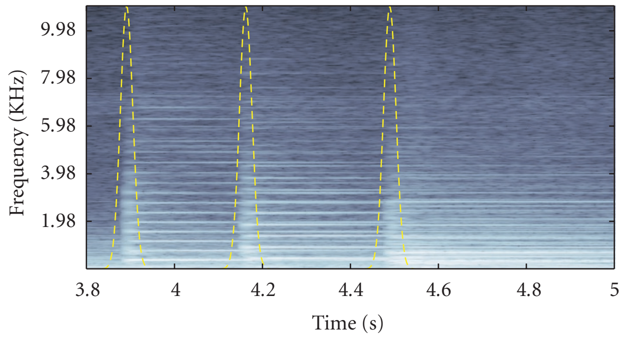

Figure 4: The magnitude plane of the STFT of a guitar recording. The sampling frequency is 22050 Hz, the window width is 30 milliseconds, and the overlapping factor is 0.9. The dashed line reveals the labeled onsets positions.

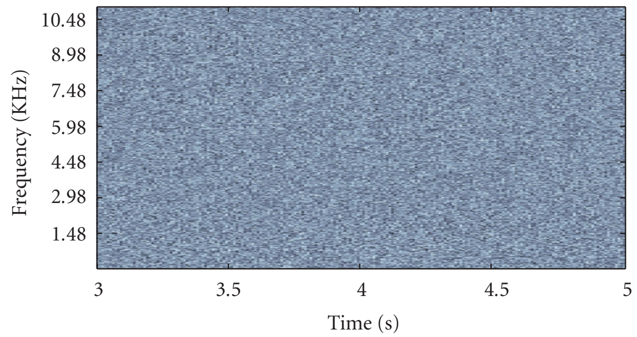

Figure 6: The phase plane of the STFT calculated in Figure 4. Unmanipulated, such a phase plane looks very much like a matrix of noise.

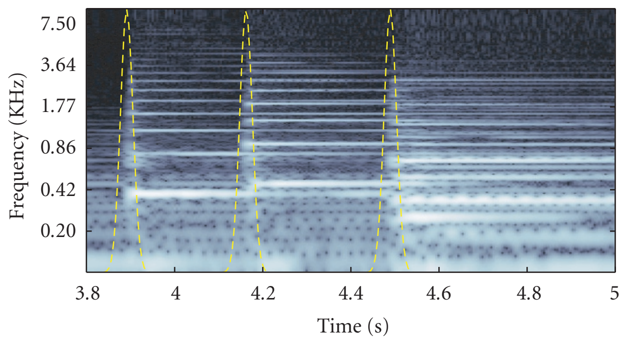

Figure 5: The magnitude plane of the constant-Q transform of the same piece as in Figure 4. The sampling frequency is 22050 Hz, the window width is 30 milliseconds, and the number of bins per octave is 48. The dashed line reveals the labeled onset positions.

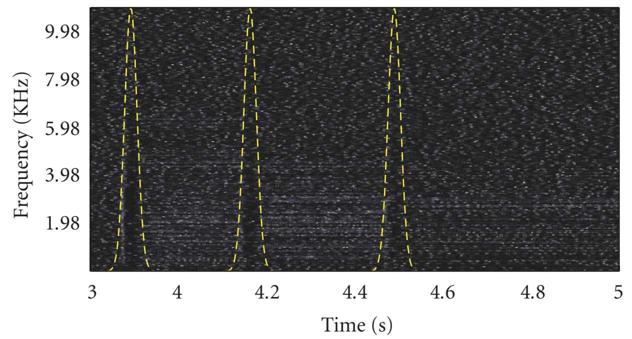

Figure 7: The phase plane of the STFT of Figure 4, transformed according to (2). The dashed line represents the labeled onsets positions. In this representation, the onset patterns are hard to see.

The logarithmic frequency scale provides a constant frequency-to-resolution ratio for a particular bin,

$$Q = \frac{f_k}{f_{k+1} - f_k} = \left(2^{1/b} - 1\right)^{-1}, \quad (5)$$

where b represents the number of bins per octave and k the frequency bin. For $b = 12$, and by choosing a particular f_0, then k is equal to the MIDI note number (which represents the equal-tempered 12-tone-per-octave scale). See Figure 5 for an example of a constant-Q transform.

As the frequency resolution is smaller at high frequencies, we can shrink the window width to yield better time resolution, which is very important for onset detection.

Like the fast Fourier transform (FFT), there is an efficient algorithm for constant-Q transform, see [14] for implementation details.

3.1.4. Phase planes

Both STFT and constant-Q are complex transforms. Therefore, we can separate their outputs into phase and magnitude planes. Obviously, the magnitude planes contain relevant information; see Figures 4 and 5. But can we do something with the phase plane? A visual observation (Figure 6) reveals that the phase plane of an STFT is quite noisy.

One potentially useful way to process the phase plane is according to (2). Experiments from [8] show that the probability distribution of phase acceleration over frequency changes significantly at the moment of a note onset. However, in some cases, these onset patterns are almost absent, as can be seen in Figure 7. Our neural network was unable to learn to find these patterns, see Table 1 for details.

So far, we have little evidence that the phase plane information differentiated along the time axis will be useful in our framework. However, the phase plane can also be differentiated along the frequency axis (i.e., columnwise rather than rowwise in the matrix),

$$\varpi_{k,n} = \operatorname{princarg}\left[\varphi_{k,n} - \varphi_{(k-1),n}\right], \quad (6)$$

where $\varpi_{k,n}$ represents the phase difference between frequency bin k and frequency bin $k - 1$ for a particular time bin n. In many cases, this yields visible patterns that correlate highly with onset times (Figure 8). This approach yields more promising results within the framework of our model. Table 1 shows that the frequency-differentiated phase plane is able to perform almost as well as the magnitude plane.

Table 1: Results for running the FNN on different kinds of representations. constant-Q performed the best, but the difference between Constant-Q and STFT is not significant. Phase acceleration did slightly better than noise, and phase difference across frequency yielded results almost as good as STFT.

Plane	Spectral window size	F-meas. train	F-meas. valid
STFT log mag	10 ms	86 ± 2	86 ± 5
STFT log mag	30 ms	86 ± 1	86 ± 5
STFT log mag	100 ms	84 ± 2	83 ± 8
C-Q log mag	10 ms	86 ± 2	86 ± 5
C-Q log mag	30 ms	**87 ± 2**	**87 ± 5**
C-Q log mag	100 ms	84 ± 2	84 ± 6
STFT ph accel	10 ms	49 ± 2	49 ± 4
STFT ph accel	30 ms	47 ± 1	47 ± 5
STFT ph accel	100 ms	49 ± 4	47 ± 6
STFT ph freq-diff	10 ms	62 ± 2	61 ± 6
STFT ph freq-diff	30 ms	80 ± 1	79 ± 4
STFT ph freq-diff	100 ms	74 ± 2	73 ± 6
Noise	—	40 ± 2	40 ± 6

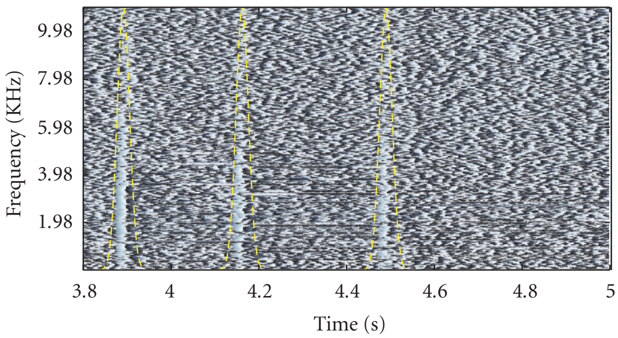

Figure 8: The phase plane of the STFT of Figure 4 transformed according to (6). The dashed line represents the labeled onsets positions.

3.2. Supervised learning for onset emphasis

We employ a feed-forward neural network (FNN) to combine evidence from the different transforms in order to classify the frames. Our goal is to use the neural net as a filtering step in order to provide the best possible trace for the peak-picking part. The network predicts the class membership (onset or nononset) of each frame in a sequence. The evidence available to the network for each prediction consists of the different spectral features extracted from the PCM signal as described above. For a given frame, the network has an access to the features for the frame in question as well as nearby frames. In this section, we use the term "window" to refer to the size of the input window defining which feature frames are fed into the FNN. (This is in contrast to the spectral window used to calculate the spectrogram in Section 3.1.1.) See Figure 9 for example.

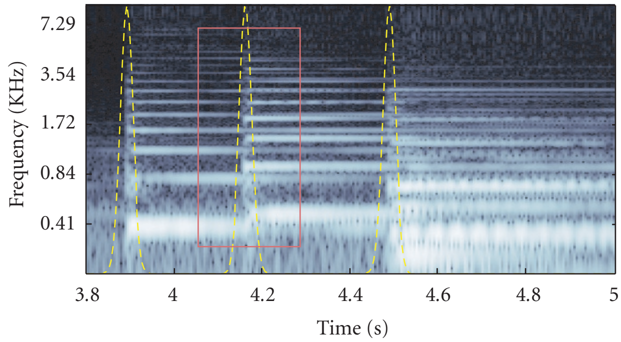

Figure 9: The constant-Q transform of a piano musical piece with labeled onsets. The dashed line is the onset trace, it corresponds to the ideal input for the peak-picking algorithm. The red box is a window seen by the neural network for a particular time and particular frequency. This input window has a width of 200 milliseconds.

3.2.1. Input variables

Onsets patterns are translation invariant on the time axis. That is, the probability distribution over all the possible patterns presented to the network does not depend on the time value,

$$p(X = x \mid T = t) = p(X = x), \quad x \in \mathbb{R}^n, \tag{7}$$

where n is the number of input variables, x represents a particular input to the network, and t is the central time of the window.

Unfortunately, the frequency axis does not exhibit this same shift invariance,

$$p(X = x \mid F = f) \neq p(X = x), \tag{8}$$

where f is the central frequency of the input window. For example, when using the STFT, an onset with a fundamental at a higher frequency will have more widely spaced harmonics than a low-frequency onset. For the case of constant-Q transform, the distances between harmonics are indeed shift invariant. However, for low frequencies, the patterns are highly blurred over frequency and time.

Despite this, a small frequency shift introduces only small changes in the underlying probability distributions,

$$|f_1 - f_2| < \epsilon \Longrightarrow p(x \mid f_1) \simeq p(x \mid f_2), \tag{9}$$

where ϵ should be positive and relatively small.

As the spectrogram is not padded, the input window can be translated only where it completely fits within the boundaries of the spectrogram. Thus, if we choose an input window height of 100% of the spectrogram height, we have no possibility for frequency translation at all. By reducing the window height to 90% of the spectrogram height (Figure 9), we are then able to make frequency translations that satisfy (9). For example, if we have 200 frequency bins, the input window will have a height of 180 frequency bins, and there will be 21 possible input window positions. For efficiency reasons, we chose only 10 evenly spaced frequency positions. The goal

TABLE 2: Results for testing different input window sizes and different numbers of input variables. Above the number of input variables is held constant at 200. Below the input window width is held constant at 300 milliseconds. It is shown that the input window width is not crucial provided that it is large enough. However, the number of input variables is important.

Input window width	No. input variables	F-meas. train	F-meas. valid
450 ms	200	86 ± 2	86 ± 6
300 ms	200	86 ± 2	86 ± 6
150 ms	200	86 ± 2	86 ± 5
75 ms	200	85 ± 2	84 ± 5
300 ms	100	84 ± 2	84 ± 6
300 ms	200	86 ± 2	86 ± 6
300 ms	400	87 ± 2	87 ± 5
300 ms	800	87 ± 2	87 ± 6

of performing translation over frequency is to have a smaller input window, thus yielding fewer parameters to learn. This strategy also provides multiple similar versions of the onset trace, yielding a more robust model.

Unfortunately, even after frequency translation, there were still too many variables in the input window to compute efficiently. To address this, we used a random sampling technique. Input window values along the frequency axis were sampled uniformly. However, sampling along the time axis was done using a normal distribution centered at the onset time. This strategy allowed us to concentrate our computational resources near the onset time. Table 2 shows results using different sampling densities. One hundred variables were insufficient for optimal performance, but any value over 200 yielded good results.

3.2.2. *Neural network structure*

Our main goal is to use a supervised approach to enhance the salience of onsets by learning from labeled examples. To achieve this, we employed a feed-forward neural network (FNN) with two hidden layers and a single neuron in the output layer. The hidden layers used tanh activation functions and the output layer used the logistic sigmoid activation function. Our choice of architecture was motivated by general observations that multihidden layer networks may offer better accuracy with fewer weights and biases than networks with single hidden layers. See Bishop [15, Chapter 4] for a discussion.

The performance for different network architectures is shown in Section 5. Table 2 shows network performance for different numbers of input variables and Table 3 shows performance for different numbers of hidden units. A typical structure uses 150 inputs variables, 18 hidden units in the first layer, and 15 hidden units in the second layer.

TABLE 3: Results from tests using different neural network architectures.

1st layer	2nd layer	F-meas. train	F-meas. valid
50	30	87 ± 2	87 ± 5
20	15	87 ± 1	87 ± 4
10	5	87 ± 2	87 ± 5
10	0	86 ± 2	86 ± 4
5	0	86 ± 2	85 ± 3
2	0	85 ± 2	85 ± 5
1	0	83 ± 2	83 ± 4

3.2.3. *Target and error function*

Recall that the goal of the network is to produce the ideal trace for the peak-picking part. Such a target trace can be a mixture of very peaked Gaussians, centered on the labeled onset time,

$$T_s(t) = \sum_i \exp^{-(\tau_{s,i}-t)^2/\sigma^2}, \tag{10}$$

where $\tau_{s,i}$ is the ith labeled onset time of signal s and σ is the width of the peak and is chosen to be 10 milliseconds.

The problem could also have been treated as a 0-1 onset/nononset classification problem. However, the abrupt transitions between onset and nononset in the 0/1 formulation proved to be more difficult to model than the smooth transitions provided by mixture of Gaussians.

For each time step, the FNN predicted the value given by the target trace. The error function is the sum of squared error over all patterns,

$$E = \sum_{s,j} \left(T_s(t_j) - O_s(t_j)\right)^2, \tag{11}$$

where $O_s(t_j)$ is the output of the network for pattern j of signal s.

3.2.4. *Learning function*

The learning function is the Polak-Ribiere version of conjugate gradient descent as implemented in the Matlab Neural Network Toolbox.

To prevent the learner from overfitting, we employed the commonly used regularization technique of *early stopping*. In early stopping, learning is terminated when performance worsens on a small out-of-sample dataset reserved for this purpose [15].

We also used cross-validation. For more details on cross-validation, see Section 5. For details on the dataset, see Section 4.

3.3. *Peak picking*

The final step of our approach involves deciding which peaks in our trace are to be treated as onsets. In our model, this peak-picking process consists of three separate operations: *merging*, *peak extraction*, and *threshold optimization*.

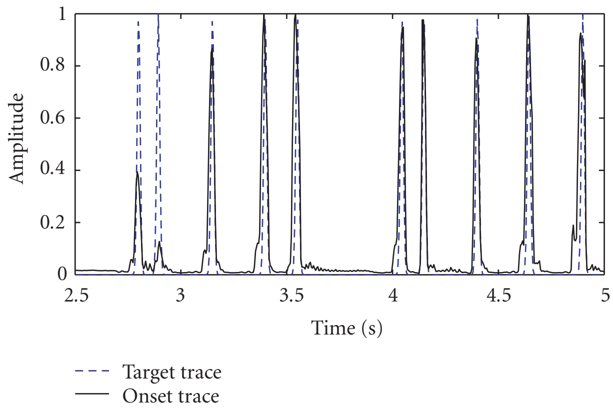

Figure 10: The target trace represents the ideal curve for the peak-picking part of the algorithm. The onset trace shows the merged output of the neural network.

3.3.1. Merging

As explained in Section 3.2.1, for reasons of robustness and efficiency, an input window is applied to the spectrogram in order to sample from a restricted range of frequencies. As this window is moved up or down in frequency, multiple sets of values for a single frame are generated. We process these sets of values individually and merge their results by averaging, generating a single onset trace, see Figure 10 for an example.

3.3.2. Peak extraction

To ensure that low-frequency trends in the signal do not distort peak height, we used a high-pass spatial filter to isolate the high-frequency information of interest (including our peaks). This high-pass filter was implemented subtractively: we cross-correlated the signal using a Gaussian filter having 500 milliseconds of standard deviation. We then subtracted this filtered version from the original signal, thus removing low-frequency trends. Finally, we set to zero all values falling below a threshold. These manipulations are expressed as follows:

$$\rho_s(t) = O_s(t) - u_s(t) + K, \quad (12)$$

where

$$u_s(t) = g * O_s(t), \quad (13)$$

where g is the Gaussian filter, K is the threshold, and ρ_s is the peak trace of signal s. Using this approach, each zero crossing with positive slope represents the beginning of an onset and each zero crossing in a negative slope represents the end of an onset.

The position of the onset is taken by calculating the center of mass of all points inside the peak,

$$\tau_{s,i} = \frac{\sum_{j \in p_i} t_j \rho_s(t_j)}{\sum_{j \in p_i} \rho_s(t_j)}, \quad (14)$$

where $\tau_{s,i}$ is the ith onset time of piece s and j is element of all the points contained in peak i.

3.3.3. Threshold optimization

To optimize performance, the value of the threshold K in (12) is learned using samples from the training set. In order to make such an optimization, we require a way to gauge the overall performance. For this, we adapt[1] the standard F-measure to our task:

$$P = \frac{n_{cd}}{n_{cd} + n_{fp}}, \qquad R = \frac{n_{cd}}{n_{cd} + n_{fn}}, \qquad F = \frac{2PR}{P + R}, \quad (15)$$

where n_{cd} is the number of correctly detected onsets, n_{fn} is the number of false negatives, and n_{fp} is the number of false positives. A perfect score gives an F-measure of 1 and for a fixed number of errors, the F-measure is optimal when the number of false positives equals the number of false negatives.

Since the peak-picking function is not continuous, we cannot use gradient descent for optimization. The optimization of noncontinuous values such as K is usually achieved using a line search algorithm like the golden section (see [16, Section 10.1]). Fortunately, we have only one parameter to optimize, thus making it possible to use a simpler method. Specifically, we carried out a grid search over 25 values of K where $0.02 \le K \le 0.5$ and retained the best performing value.

3.4. MULTI-NET variant

Our exploration of input representations and neural network architectures led us to the conclusion that there was no optimal set of hyperparameters for our SINGLE-NET model. In an attempt to increase model robustness, we decided to test a simple ensemble learning approach by combining the results of several SINGLE-NET learners trained with different hyperparameters on the same dataset. In this section, we describe the details of the resulting MULTI-NET model.

For the simulations described here, a MULTI-NET consists of seven SINGLE-NET networks trained using different hyperparameters. In addition, the SINGLE-NET networks each benefited from a tempo trace calculated using predicted onsets. An additional FNN was used to mix the results and to derive a single prediction.

In raw performance terms, the additional complexity of MULTI-NET seems warranted. For example, in the MIREX 2005 Contest (described briefly in Section 5.1), MULTI-NET outperformed SINGLE-NET by 1.7% of F-measure and won the first place. Details of the two major parts of MULTI-NET, the tempo-trace computation and the merging procedure, are explained in the following sections.

[1] This F-measure was also used in the MIREX 2005 Audio Onset Detection Contest.

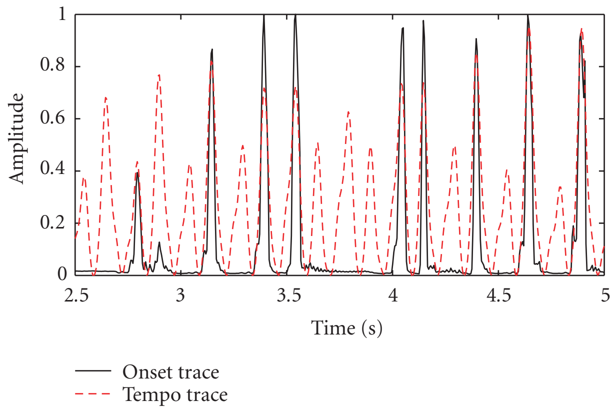

Figure 11: The onset trace shows the merged output of the neural networks as in Figure 10. The tempo trace shows the cross-correlation of the onset trace with its own autocorrelation.

3.4.1. Tempo trace

The SINGLE-NET variant has access only to short-timescale information available from near-neighbor frames. As such, it is unable to discover regularities that exist at longer timescales. One important regularity is tempo. The rate of note production is useful for predicting note onsets. For the MULTI-NET variant, we calculate a tempo trace that can be used to condition the probability that a particular point in time is an onset.

To achieve this, we compute the tempo trace Γ by correlating the interonset histogram of a particular point in the onset trace with the inter-onset histogram of all other onsets. If the two histograms are correlated, this indicates that this point is in phase with the tempo,

$$\Gamma(t) = h\Big(\{\mu_i - \mu_j\}_{ij}\Big) \cdot h\Big(\{\mu_i - t\}_i\Big), \tag{16}$$

where $\Gamma(t)$ is the tempo trace at time t, $h(S)$ is the histogram of set S, and μ_i is the ith onset. The dot product between the two histograms is the measure of correlation.

This method calculates n histograms, with each of them requiring time $O(n)$ to compute. Therefore, the algorithm is $O(n^2)$. Moreover, if errors occur in the peak extraction, they directly affect the results of these histograms. To compensate for this, Section 3.5 introduces a way to calculate the tempo trace directly on the onset trace by computing the cross-correlation of the onset trace with the onset trace's autocorrelation. This yields an algorithm with complexity $O(n \log n)$, see Figure 11 for an example.

3.4.2. Tempo-trace confidence

The tempo trace allows the final FNN to perform categorization based not only on the ambiguity of a peak but also on whether we are *expecting* a peak or not at this particular time. In addition, we provide the network with the normalized entropy of the interonset histogram as a measure of rhythmicity,

$$H(T) = \frac{1}{\log_2 n} \sum_{i=1}^{n} p(t_i) \log_2 p(t_i), \tag{17}$$

where the normalization factor serves to map every measure of entropy between 0 and 1. This provides the network with a measure of confidence when weighing the relative influence of the tempo.

3.4.3. Merging information

In order to merge information for the MULTI-NET variant of our approach, we simply stack all the onset traces from our multiple networks along with their tempo traces (including the entropy-based prediction about rhythmicity). For example, the 10 frequency translations with the onset trace and the rhythmicity yield 12 traces per model. Using 7 models gives a matrix of 84 rows.

This merged information yields a matrix with a sampling rate equal to the original spectrogram, but containing different information. We continue with the SINGLE-NET variant using this new feature frame in place of the original spectrogram. Unlike the SINGLE-NET variant, the input window takes into account 100% of the frequency spectrum. That is, no sliding window over frequency is used because there is no longer any continuity over frequency in the features we extracted.

3.5. Tempo trace by autocorrelation

In this section, we review autocorrelation and tempo induction. We then show that (16) can be calculated directly on the onset trace by cross-correlating the signal with the autocorrelation of the same signal.

3.5.1. Autocorrelation and tempo

The autocorrelation of a signal provides a high-resolution picture of the relative salience of different periodicities, thus motivating its use in tempo- and meter-related music tasks. However, the autocorrelation transform discards all phase information, making it impossible to *align* salient periodicities with the music. Thus autocorrelation can be used to predict, for example, that music has something that repeats every 1000 milliseconds but it cannot say *when* the repetition takes place relative to the start of the music.

Autocorrelation is certainly not the only way to compute a tempo trace. Adaptive oscillator models [17, 18] can be thought of as a time-domain correlate to autocorrelation based methods and have shown promise, especially in cognitive modeling. The integrate-and-fire neural network from [12] can be viewed as such an oscillator-based approach. Multiagent systems such as those by Dixon [19] have been applied with success, as have Monte Carlo sampling [20] and Kalman filtering methods [21].

Many researchers have used autocorrelation to find tempo in music. Brown [22] was perhaps the first to use autocorrelation to find temporal structure in musical scores.

Scheirer [2] extended this work by treating audio files directly. Tzanetakis and Cook [23] used autocorrelation to generate a beat histogram as a feature for music classification. They perform peak-picking as part of computing the beat histogram, whereas peak-picking is our primary goal here. Both Toiviainen and Eerola [24] and Eck [25] used autocorrelation to predict the meter in musical scores. Klapuri et al. [4] incorporated the signal processing approaches of Goto [26] and Scheirer in a model that analyzes the period and phase of three levels of the metrical hierarchy. Eck [27] introduced a method that combines the computation of phase information and autocorrelation so that beat induction and tempo prediction could be done directly in the autocorrelation framework.

3.5.2. *Tempo trace by autocorrelation*

We will now prove that a tempo trace based on interonset histograms can be calculated via autocorrelation. To start, let us assume that the interonset histogram is equal to the autocorrelation of the onset trace (in fact this is the case, as is shown below),

$$h_a(t) = \gamma \star \gamma, \tag{18}$$

where $h_a(t)$ is the interonset histogram for interonset time t, γ is the original onset trace, and $\star$ is the cross-correlation operator. Using this to rewrite (16) gives

$$\begin{aligned}\Gamma(t) &= \int h_a(t'')(\gamma \star \delta_t)dt'' \\ &= \int h_a(t'')\Big(\int \gamma(t')\delta(t' - t + t'')dt'\Big)dt'' \\ &= \int h_a(t'')\gamma(t + t'')dt'' = (\gamma \star \gamma) \star \gamma,\end{aligned} \tag{19}$$

where $\Gamma(t)$ is the tempo trace at time t and $\delta_t \equiv \delta(\tau - t)$, where δ is the delta Dirac.

Therefore, the tempo trace can be calculated by correlating the onset trace 3 times with itself. This operation takes now time $O(n \log n)$, which is much faster than the $O(n^2)$ required by (16).

3.5.3. *Interonset histogram by autocorrelation*

What remains is to demonstrate that the interonset histogram of a peaked trace is in fact equal to the autocorrelation of a peaked trace. To achieve this, we first show that the autocorrelation of the sum of a function is the pairwise cross-correlation of all functions,

$$\begin{aligned} f(t) &\equiv \sum_i g_i(t), \\ f(t) \star f(t) &= \mathfrak{F}\big[|F(k)|^2\big] = \mathfrak{F}\Big[\sum_{ij} \overline{G_i(k)}G_j(k)\Big] \\ &= \sum_{ij} g_i(t) \star g_j(t),\end{aligned} \tag{20}$$

where $F(k)$ and $G_i(k)$ are, respectively, the results of the Fourier transform of $f(t)$ and $g_i(t)$. $\mathfrak{F}$ is the Fourier transform operator.

It is a known result that the cross-correlation of two Gaussians is another Gaussian with the new mean given by $\mu_1 - \mu_2$ and the new variance is $\sigma_1^2 + \sigma_2^2$,

$$\begin{aligned} N(t;\mu_1,\sigma_1) &\star N(t;\mu_2,\sigma_2) \\ &= N\Big(t;(\mu_1-\mu_2),\sqrt{\sigma_1^2+\sigma_2^2}\Big),\end{aligned} \tag{21}$$

where

$$N(t;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}}e^{-(t-\mu)^2/\sigma^2}. \tag{22}$$

If we approximate the onset trace as being a mixture of Gaussians

$$\gamma(t) = \sum_i \alpha_i N(t;\mu_i,\sigma_i), \tag{23}$$

then, using (20) and (23), we can rewrite the autocorrelation of the onset traces

$$\gamma(t) \star \gamma(t) = \sum_{ij} \big(\alpha_i N(t;\mu_i,\sigma_i)\big) \star \big(\alpha_j N(t;\mu_j,\sigma_j)\big) \tag{24}$$

and with (21), (24) becomes

$$\sum_{ij} \alpha_i\alpha_j N\Big(t;(\mu_i-\mu_j),\sqrt{\sigma_i^2+\sigma_j^2}\Big), \tag{25}$$

which is a more general case of a Parzen window histogram. The traditional case is where α_i and σ_i remain constant across points. This loss of information occurs when we extract the peaks from the onset trace, keeping only the position and ignoring the width and the height.

4. DATASET

To learn this task correctly, we needed a dataset with accurate annotations that covers a wide variety of musical styles. Accuracy is particularly important for this task because temporal errors in mislabeling will have grave effects: the network will be punished for predicting an onset at the correct position and will be punished for *not* predicting an onset at the erroneous position.

The most promising candidate dataset we found was a publicly available collection from Leveau et al. [28]. Unfortunately, this dataset was too small and restricted for our purposes, mainly focusing on monophonic pieces.

We chose to annotate our own musical pieces. To make it possible to share our annotations with others, we selected the publicly available nonannotated "Ballroom" dataset from ISMIR 2004 as a source for our waveforms. The "Ballroom" dataset is composed of 698 wav files of approximately 30 seconds each. Annotating the complete dataset would be too time consuming and was not necessary to train our model. We therefore annotated 59 random segments of 10 seconds each. Most of them are complex and polyphonic with singing, mixed with pitched and noisy percussions.

The labels were manually annotated using a Matlab program with GUI constructed by the first author to allow for precise annotation of wav files. The "Ballroom" annotations as well as the Matlab interface are available on request from the first author or at the following page: http://www-etud.iro.umontreal.ca/~lacostea

5. RESULTS

To choose among different methods and different hyperparameters, we tested the SINGLE-NET algorithm using 3 fold cross-validation on the "Ballroom" dataset (Section 4). 15 pieces out of 69 were used for the test set and the 3 different separations yield a measure of variance for both the training and tests results.

A typical spectrogram contains 200 frames per second, and each piece lasts 10 seconds. Taking into account the 10 frequency translations, this yields 20 000 input patterns per piece. Learning from all of these patterns is redundant and prohibitively slow. Thus we use only 5% of them, yielding a total of 54 000 training examples. This in practice was demonstrated to be enough data to prevent overfitting. The dataset had an imbalanced ratio of onsets and nononsets (positive and negative examples). In early training runs, we tried sampling preferentially from frames near onsets. This had no noticeable effect in the behavior of the model so for later learning runs, including those discussed here, we did not balance the training data.

For those tests, parameters not specified are assumed to be the default as specified here: input window size is 150 milliseconds, sampling rate is 200 Hz, number of input variables is 150, number of hidden units in layer one is 18, number of hidden units in layer two is 15, and the Hamming window size is 30 milliseconds.

The first test we made is to determine which plane is appropriate for detecting onsets. We tested the logarithm of the magnitude of the STFT, the logarithm of the amplitude of the constant-Q transform, the phase acceleration, and the phase difference along the frequency axis. For each of these, we evaluated model performance for different window widths. Table 1 shows the results for these tests. The best performance was achieved with the constant-Q transform, but the difference between constant-Q and STFT is not significant. The exact window width is not crucial provided it is small enough. The phase acceleration performed only slightly better than noise; however, the phase difference along frequency axis worked much better, performing almost as well as the STFT magnitude plane.

We then evaluated the input window width and the number of input variables on the magnitude plane of the STFT. Table 2 shows that the input window width size is not crucial provided that it is not too small. However, the number of input variables is indeed important, with saturation occurring at around 400.

In Table 3, we report performance results for different network architectures. It can be seen that networks with two hidden layers perform better than those having only a single hidden layer. Also, it can bee seen that a relatively small number of neurons is sufficient for good performance (10 and 5 for the first and second layers, resp.). It is also interesting to note that a single neuron performs reasonably well (F-measure of 83 versus 87 for our best performing model). This suggests that it may be possible to construct a simple, highly efficient version of our model that can work on very large datasets.

Table 4: Results from tests combining STFT log-magnitude plane with the phase difference across frequency plane as input to the network. Unfortunately, the addition of phase difference in the frequency axis does not yield better results than the STFT log magnitude alone.

No. input variables	Hamming window size	F-meas. train	F-meas. valid
100	30 ms	85 ± 2	84 ± 5
100	50 ms	85 ± 1	84 ± 7
100	100 ms	80 ± 2	79 ± 8
200	30 ms	86 ± 2	86 ± 5
200	50 ms	86 ± 2	85 ± 6
200	100 ms	84 ± 2	84 ± 7

Table 5: Overall results of the MIREX 2005 onset detection contest for our two variants. Their F-measures were the two highest. They also had the best balance between the precision and recall. This is probably due to to the learned threshold in the peak-picking part.

Variant	MULTI-NET	SINGLE-NET
Overall average F-measure	80.07%	78.35%
Overall average precision	79.27%	77.69%
Overall average recall	83.70%	83.27%
Total correct	7974	7884
Total false positives	1776	2317
Total false negatives	1525	1615
Total merged	210	202
Total doubled	53	60
Runtime(s)	4713	1022

Table 1 suggests that combining the magnitude plane with the phase plane might yield better results. In Table 4, we report results from testing this idea using different numbers of input variables and different Hamming window sizes. In the table, the number of input variables corresponds to the number of points for each plane. Unfortunately, the combination of magnitude plane with phase plane does not yield better results.

5.1. MIEX 2005 results

Both variants of our algorithm were entered in the MIREX 2005 Audio Onset Detection Contest. The MIREX 2005 dataset is composed of 30 solo drum pieces, 30 solo monophonic pitched pieces, 10 solo polyphonic pitched pieces, and 15 complex mixes. On this dataset, the MULTI-NET algorithm performed slightly better than the SINGLE-NET algorithm. MULTI-NET yielded an F-measure of 80.07% while SINGLE-NET yielded an F-measure of 78.35% (see Table 5). These results yielded the best and second best performance, respectively, for the contest. See Table 6 for results.

Table 6: Overall scores from the MIREX 2005 audio onset detection contest. Overall average F-measure, overall average precision, and overall average recall are weighted by number of files in each of nine classes.

Rank	Participant	Avg. F-measure	Avg. precision	Avg. recall
1	Lacoste & Eck (MULTI-NET)	80.07%	79.27%	83.70%
2	Lacoste & Eck (SINGLE-NET)	78.35%	77.69%	83.27%
3	Ricard, J.	74.80%	81.36%	73.70%
4	Brossier, P.	74.72%	74.07%	81.95%
5	Röbel, A. (2)	74.64%	83.93%	71.00%
6	Collins, N.	72.10%	87.96%	68.26%
7	Röbel, A. (1)	69.57%	79.16%	68.60%
8	Pertusa, Klapuri, & Iñesta	58.92%	60.01%	61.62%
9	West, K.	48.77%	48.50%	56.29%

Table 7: F-measure percentages for all nine classes from the MIREX 2005 audio onset detection contest. Best performance for each class is shown in bold. The number of pieces for each class is shown in parentheses.

	Complex (15)	Poly-pitched (10)	Bars and bells (4)	Brass (2)	Drum (30)	Plucked string (9)	Singing voice (5)	Sust. strings (6)	Wind (4)
MULTI-NET	**78.85**	**86.31**	86.55	70.25	91.40	81.84	**45.33**	56.68	58.75
SINGLE-NET	77.02	85.93	86.37	67.88	89.91	83.49	34.35	52.87	56.48
Ricard, J.	71.90	83.26	87.17	**72.66**	90.97	77.85	27.59	38.45	38.57
Brossier, P.	76.16	80.88	73.97	64.88	86.28	79.99	22.16	**57.92**	52.08
Röbel, A. (2)	62.84	76.24	90.34	68.32	89.96	**84.20**	40.68	36.18	**66.01**
Collins, N.	60.25	75.70	**99.28**	69.09	**92.31**	81.97	29.34	14.74	47.57
Röbel, A. (1)	59.76	69.29	97.92	61.87	86.29	77.58	42.69	17.35	51.13
Pertusa et al.	50.16	59.37	60.22	54.41	77.22	67.74	11.12	38.45	25.59
West, K.	47.13	39.98	34.58	33.94	71.61	39.85	12.07	32.12	18.11

Both variants of the algorithm were designed to perform well on a wide range of music, so they were less efficient than other algorithms on monophonic pieces. But when all pieces are considered, MULTI-NET and SINGLE-NET were the two best-performing entries in the contest. Both variants also showed a good balance between precision and recall. This advantage is likely due to the learned threshold in the peak-picking part (Section 3.3).

6. DISCUSSION

An in-depth analysis of model errors on the annotated "Ballroom" dataset shows that most of the false negatives are produced by pitched onsets with thin harmonics. This is surprising because such onsets are easily perceived by human. Our failure here is likely due to the fact that we only pick a random subset of the variables from the input window. Picking more variables helps, but for some pitched sounds so few variables are responsible for coding the onset that the FNN still fails. Incidentally, this perhaps explains why our entry performed poorly on the category solo bars and bells (see Table 7). We had an F-measure of 86.55% where the best for this category was 99.28%. False positives, on the other hand, were mainly generated by singing or vibrato, as expected. But the algorithm is still quite robust for those events.

There are also some boundary effects. At the beginning and at the end of sequences, the network was often unable to adequately resolve onsets. One solution to this problem could be to train three different networks, one that predicts onset using only information from the past, a second that uses only information from the future, and a third one (like the current model) that incorporates past and future frames. For the first few frames, we could use the "future-only" version, for the last frames, the "past-only" model and for all other frames the "past-future" version. Moreover, the causal "past-only" version could also be used for online detection.

On a more general note, we mentioned several tasks that might benefit from good audio onset detection, such as tempo detection, classification, and fingerprinting. This is not to say that onset detection is required for tasks like these. In fact, the MIR community seems mixed on the usefulness of onset detection in this domain: of the 13 entries in the MIREX 2005 Tempo Contest, only 4 of them used detected onsets or onset energy functions [29]. This may be due to a philosophical rejection of onset detection as a part of tempo finding. Scheirer [2] argued, for example, that explicit note detection was not evident in the auditory system and not necessary for tempo and beat analysis. However, it could also be simply due to the fact that onset detection algorithms have, to date, not worked very well. For example, this was the main reason that the second author of this paper did not use an onset detector in his MIREX entry [30].

6.1. Future work

Though our results are relatively good, there is still much room for improvement. The ability to perform good pitch detection would definitively improve model performance for notes that have thin harmonics. Another way would be to train a second network on a dataset of pitched onsets.

Different kinds of machine learning approaches can also be used for this problem. Convolutional networks [31] would be able to use a wider window and take advantage of all input variables while still employing a reasonable amount of parameters.

Working on a low-dimensional set of features instead of the entire spectrogram could provide speed improvements and could yield good results with a lower-capacity network. This would allow us to train on a much larger annotated dataset, perhaps yielding better generalization.

7. CONCLUSIONS

We have presented an algorithm that adds a supervised learning step to the basic onset detection framework of signal transformation, feature enhancement, and peak picking. Our SINGLE-NET variant used a single feed-forward neural network to enhance spectrogram frames for peak picker. Our MULTI-NET variant combined the predictions of several SINGLE-NET networks with tempo traces to improve performance. Though both models show promise, we believe that the SINGLE-NET model warrants more attention due to its relative simplicity. We provided evidence that our algorithm works well, comparing it positively with other state-of-the-art approaches. We conclude that the general approach of supervised learning makes sense in the domain of audio note onset detection.

APPENDIX

SUMMARY OF MIREX 2005 AUDIO ONSET DETECTION RESULTS

The goal of the contest was to evaluate and compare onset detection algorithms applied to audio music recordings. The dataset consisted of 85 audio files (14.8 minutes total) from 9 classes: complex, polypitched, solo bars and bells, solo brass, solo drum, solo plucked strings, solo singing voice, solo sustained strings, and solo winds. This information is summarized from http://www.music-ir.org/evaluation/mirex-results/audio-onset/index.html

REFERENCES

[1] K. West and S. Cox, "Finding an optimal segmentation for audio genre classification," in *Proceedings of 6th International Conference on Music Information Retrieval (ISMIR '05)*, pp. 680–685, London, UK, September 2005.

[2] E. D. Scheirer, "Tempo and beat analysis of acoustic musical signals," *Journal of the Acoustical Society of America*, vol. 103, no. 1, pp. 588–601, 1998.

[3] A. Klapuri, "Sound onset detection by applying psychoacoustic knowledge," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '99)*, vol. 6, pp. 3089–3092, Phoenix, Ariz, USA, March 1999.

[4] A. P. Klapuri, A. J. Eronen, and J. T. Astola, "Analysis of the meter of acoustic musical signals," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 1, pp. 342–355, 2006.

[5] F. Gouyon, A. Klapuri, S. Dixon, et al., "An experimental comparison of audio tempo induction algorithms," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 5, pp. 1832–1844, 2006.

[6] C. Duxbury, J. P. Bello, M. Davies, and M. Sandler, "Complex domain onset detection for musical signals," in *Proceedings of 6th International Conference on Digital Audio Effects (DAFx '03)*, London, UK, September 2003.

[7] C. Duxbury, J. P. Bello, M. Davies, and M. Sandler, "A combined phase and amplitude based approach to onset detection for audio segmentation," in *Proceedings of the 4th European Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS '03)*, London, UK, April 2003.

[8] J. P. Bello and M. Sandler, "Phase-based note onset detection for music signals," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '03)*, vol. 5, pp. 441–444, Hong Kong, April 2003.

[9] J. P. Bello, C. Duxbury, M. Davies, and M. Sandler, "On the use of phase and energy for musical onset detection in the complex domain," *IEEE Signal Processing Letters*, vol. 11, no. 6, pp. 553–556, 2004.

[10] E. Kapanci and A. Pfeffer, "A hierarchical approach to onset detection," in *Proceedings of the International Computer Music Conference (ICMC '04)*, Miami, Fla, USA, October 2004.

[11] M. Davy and S. Godsill, "Detection of abrupt spectral changes using support vector machines an application to audio signal segmentation," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '02)*, vol. 2, pp. 1313–1316, Orlando, Fla, USA, May 2002.

[12] M. Marolt, A. Kavcic, and M. Privosnik, "Neural networks for note onset detection in piano music," in *Proceedings of the International Computer Music Conference (ICMC '02)*, Gotenborg, Sweden, September 2002.

[13] J. C. Brown, "Calculation of a constant Q spectral transform," *Journal of the Acoustical Society of America*, vol. 89, no. 1, pp. 425–434, 1991.

[14] J. C. Brown and M. S. Puckette, "An efficient algorithm for the calculation of a constant Q transform," *Journal of the Acoustical Society of America*, vol. 92, no. 5, pp. 2698–2701, 1992.

[15] C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, UK, 1995.

[16] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, Cambridge, Mass, USA, 2nd edition, 1993.

[17] E. W. Large and J. F. Kolen, "Resonance and the perception of musical meter," *Connection Science*, vol. 6, no. 1, pp. 177–208, 1994.

[18] D. Eck, "Finding downbeats with a relaxation oscillator," *Psychological Research*, vol. 66, no. 1, pp. 18–25, 2002.

[19] S. E. Dixon, "Automatic extraction of tempo and beat from expressive performances," *Journal of New Music Research*, vol. 30, no. 1, pp. 39–58, 2001.

[20] A. T. Cemgil and B. Kappen, "Monte Carlo methods for tempo tracking and rhythm quantization," *Journal of Artificial Intelligence Research*, vol. 18, pp. 45–81, 2003.

[21] A. T. Cemgil, B. Kappen, P. W. M. Desain, and H. J. Honing, "On tempo tracking: tempogram representation and Kalman filtering," *Journal of New Music Research*, vol. 29, no. 4, pp. 259–273, 2001.

[22] J. C. Brown, "Determination of the meter of musical scores by autocorrelation," *Journal of the Acoustical Society of America*, vol. 94, no. 4, pp. 1953–1957, 1993.

[23] 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.

[24] P. Toiviainen and T. Eerola, "The role of accent periodicities in meter induction: a classificatin study," in *Proceedings of the 8th International Conference on Music Perception and Cognition (ICMPC8 '04)*, S. Lipscomb, R. Ashley, R. Gjerdingen, and P. Webster, Eds., Causal Productions, Evanston, Ill, USA, August 2004.

[25] D. Eck, "A machine-learning approach to musical sequence induction that uses autocorrelation to bridge long timelags," in *Proceedings of the 8th International Conference on Music Perception and Cognition (ICMPC8 '04)*, S. D. Lipscomb, R. Ashley, R. O. Gjerdingen, and P. Webster, Eds., Causal Productions, Evanston, Ill, USA, August 2004.

[26] M. Goto, "An audio-based real-time beat tracking system for music with or without drum-sounds," *Journal of New Music Research*, vol. 30, no. 2, pp. 159–171, 2001.

[27] D. Eck, "Meter and autocorrelation," in *10th Rhythm Perception and Production Workshop (RPPW '05)*, Blitzen, Belgium, July 2005.

[28] P. Leveau, L. Daudet, and G. Richard, "Methodology and tools for the evaluation of automatic onset detection algorithms in music," in *Proceedings of 5th International Conference on Music Information Retrieval (ISMIR '04)*, Barcelona, Spain, October 2004.

[29] M. McKinney and D. Moelants, "Mirex 2005: tempo contest," in *Proceedings of 6th International Conference on Music Information Retrieval (ISMIR '05)*, London, UK, September.

[30] D. Eck and N. Casagrande, "A tempo-extraction algorithm using an autocorrelation phase matrix and shannon entropy," MIREX tempo extraction contest, 2005, http://www.music-ir.org/evaluation/mirex-results/.

[31] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time-series," in *The Handbook of Brain Theory and Neural Networks*, M. Arbib, Ed., MIT Press, Cambridge, Mass, USA, 1995.

Alexandre Lacoste received a B.S. degree in physics and computer science (2004) from the University of Montreal, where he is currently pursuing an M.S. degree in computer science. His specialization is music and machine learning, with a focus on audio feature extraction and signal processing, supervised learning, and web-based music information retrieval.

Douglas Eck completed a Ph.D. degree in computer science and cognitive science at Indiana University (2000). He is now an Assistant Professor in the Department of Computer Science at the University of Montreal. He is also an Active Member of Brain Music and Sound BRAMS, an interdisciplinary group uniting music and brain researchers from around Montreal. His primary area is machine learning in the domain of music, with focus on areas such as rhythm and meter, music performance dynamics, and musical similarity in digital audio.