Indian Institute of Technology, Kanpur
SURGE 2019 PROJECT REPORT

Score Following: Audio to Score Alignment

Submitted by:
Nitish Vikas Deshpande
Department of Electrical Engineering, IITK

Under the guidance of:
Prof. Vipul Arora
Department of Electrical Engineering, IITK

Contents

1 Abstract

The main objective of this work is to study various techniques for the score following problem and make a comparison between the conventional methods[1] and state of the art methods for the problem. The problem consists of 2 components: Feature extraction from audio and alignment algorithms. We have chosen the state of the art algorithms for both of the components and merged them to implement an accurate score following system which has been evaluated on Bach10 as well as Altered Bach10 dataset. Due to advancements in deep learning, autoencoder networks have been used to extract features from audio rather than taking a simple transform like STFT and CQT. The incorporation of a deep learning method over the standard feature extraction procedure highly improves the accuracy of the alignment. Moreover variants of DTW like ShapeDTW which take into account the local temporal features by comparing local subsequences instead of point wise comparison and FastDTW which reduces the time complexity have been incorporated in the final implementation. This accurate alignment system forms a base for applications like music tutoring system.

Keywords Score Following, DTW, Deep learning, autoencoders, audio features, Bach10

[1] DTW with chromagrams [1]

2 Acknowledgements

I am deeply indebted to my project supervisor, Prof. Vipul Arora for his valuable guidance, fruitful discussion, constant support and encouragement throughout my SURGE program and for inculcating in me the spirit of project work. I am thankful to him for all the efforts he has taken up to help me develop my writing and presentation skills.

Further, I express my sincere gratitude to Surge program Prof. In charge Prof. Sudhir Kamle for constant support throughout the SURGE program.

I extend my acknowledgement to my friends Rohin, Akash, Shashank, Dileep and Prajwal who are a part of the MIR research group of Prof Vipul Arora. The discussions and interactions with them helped me clarify a lot of concepts. My deepest appreciation belongs to my family. Without their love and support over the year none of this would have been possible.

3 Introduction

3.1 Background of Music Information Retrieval

The objective of Music Information Retrieval research is to provide efficient mechanisms for analyzing musical material (represented either on the symbolic or the performance level), so as to derive meaningful information that is useful for a number of user applications relating to music access, music performance or music composition.

Two major constitutions in MIR today are the International Society for Music Information Retrieval (ISMIR) and the Music Information Retrieval Evaluation eXchange (MIREX). ISMIR is an international forum on MIR research topics, which holds an annual conference since 2000. Its purpose is the gathering of researchers, educators, students and professionals and the presentation and exchange of the latest news, ideas and results on MIR. MIREX is the annual contest that is taking place along with ISMIR, where researchers propose their approaches to MIR tasks

3.2 Music Terminologies

A piece of music, especially Western classical music, is often associated with multiple versions or representations, including sheet music of different editions, symbolic representations(score) of various types such as MIDI (Musical Instrument Digital Interface) and MusicXML(Music Extensible Markup Language), or audio recordings of different performances. A score is a symbolic representation of music. MIDI is one of the most popular choices for saving and editing digital music in the symbolic form. MIDI standard specifies a protocol for communication between electronic instrument. MIDI carries the control data that encodes musical performance information such as the start and stop time of a note, its pitch and its velocity as well as clock messages for synchronisation purposes. In this work we use the MIDI representations for ground truth annotations of the pitch and note onset timings.

3.3 Problem Statement

Audio-to-score alignment (also known as score following) is the process of temporally fitting music performance audio to its score. The goal is to match each musical event of a given musical score to a point in a given recording of it, thus aligning, or synchronizing the audio with the score.

3.4 General Approach to the problem

There are two main components in audio-to-score alignment task:

- Features used in comparing audio to score. There are 2 approaches for comparing audio to score:
  - Converting MIDI score to synthesized audio and comparing it to performance audio using various audio features.
  - Convert the performance audio to MIDI using automatic music transcription (AMT) systems and comparing the performance to score in the MIDI domain.

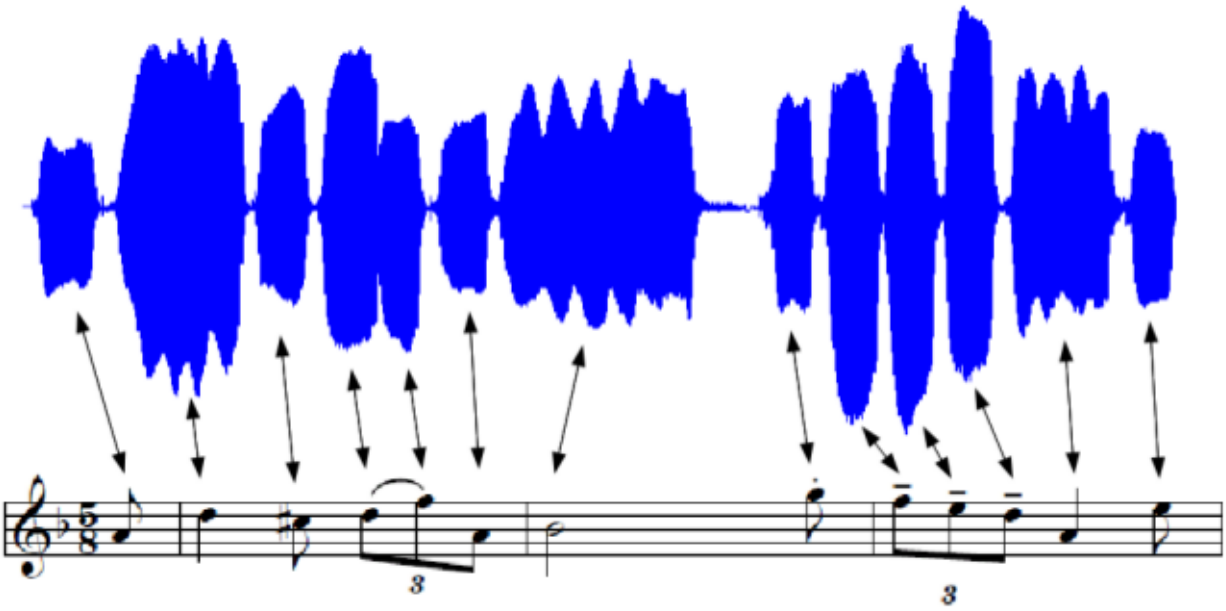

Figure 1: Assignment of each note of the score to a specific part of the performance waveform

- Alignment algorithm between two feature sequences

As automatic transcription methods are not so accurate, we will be using the first approach of converting MIDI score to synthesized audio, thus reducing the problem to audio-to-audio alignment. Study of various audio features and time-frequency representations and their comparison is described in detail in section 4.1

4 Literature Review

4.1 Study of audio features

As discussed in section 3.4, the time frequency representations which are generally used are:

- Short Time Fourier Transform (STFT)
- Constant Q transform (CQT)
- Mel Frequency Cepstral Coefficients (MFCC)

We will briefly discuss the theory of these time frequency representations, the various tuning parameters involved and the time-frequency resolution tradeoff.

4.1.1 STFT

Procedure to calculate STFT:

- Segment signal into narrow time intervals (i.e., narrow enough to be considered stationary[2])
- Take the Fourier Transform of each segment

Each Fourier Transform provides the spectral information of a separate time-slice of the signal, providing simultaneous time and frequency information. The mathematical definition of STFT is:
$X_{STFT}[m,n] = \sum_{k=0}^{L-1} x[k]g[k-m]e^{-j2\pi nk/L}$
where $x[k]$ denotes a signal and g[k] denotes an L point window function.

4.1.2 Constant Q transform

Like the Fourier transform a constant Q transform [2] is a bank of filters, but in contrast to the former it has geometrically spaced center frequencies $f_k = f_0 \cdot 2^{\frac{k}{b}} (k = 0, \ldots$ where b dictates the number of filters per octave. One chooses the bandwidth of the k-th fillter as $\Delta_k^{\text{cq}} = f_{k+1} - f_k = f_k \left(2^{\frac{1}{b}} - 1\right)$. This yields a constant ratio of frequency to resolution $Q = \frac{f_k}{\Delta_k^{cq}} = \left(2^{\frac{1}{b}} - 1\right)^{-1}$

A nice feature of this transform is that it mirrors the human auditory system, whereby at lower-frequencies spectral resolution is better, whereas temporal resolution improves at higher frequencies.

Procedure to calculate CQT :

- Choose minimal frequency f_0 and the number of bins per octave b according to the requirements of the application
- $K := b \cdot \log_2 \left(\frac{f_{\max}}{f_0}\right)$
- $Q := \left(2^{\frac{1}{b}} - 1\right)^{-1}$

[2]Non varying frequency

- $N_k := Q\frac{f_s}{f_k}$
- $x^{\text{cq}}[k] := \frac{1}{N_k}\sum_{n<N_k} x[n]w_{N_k}[n]e^{-2\pi inQ/N_k}$

4.1.3 Spectogram

A spectrogram is a visual representation of the spectrum of frequencies of a signal as it varies with time. When applied to an audio signal, spectrograms are sometimes called sonographs, voiceprints, or voicegrams. A spectrogram is usually depicted as a heat map, i.e., as an image with the intensity shown by varying the colour or brightness. A common format is a graph with two geometric dimensions: one axis represents time, and the other axis represents frequency; a third dimension indicating the amplitude of a particular frequency at a particular time is represented by the intensity or color of each point in the image.

Parameters of the spectogram include:

- Window length M: Controls frequency resolution
- Window type (Hamming, Kaiser, etc.): Controls side-lobe suppression (at the expense of resolution when M is fixed)
- Hop-size R Determines how much oversampling there will be along the time dimension.
- FFT length N Determines how much spectral oversampling (interpolation) is to be provided.

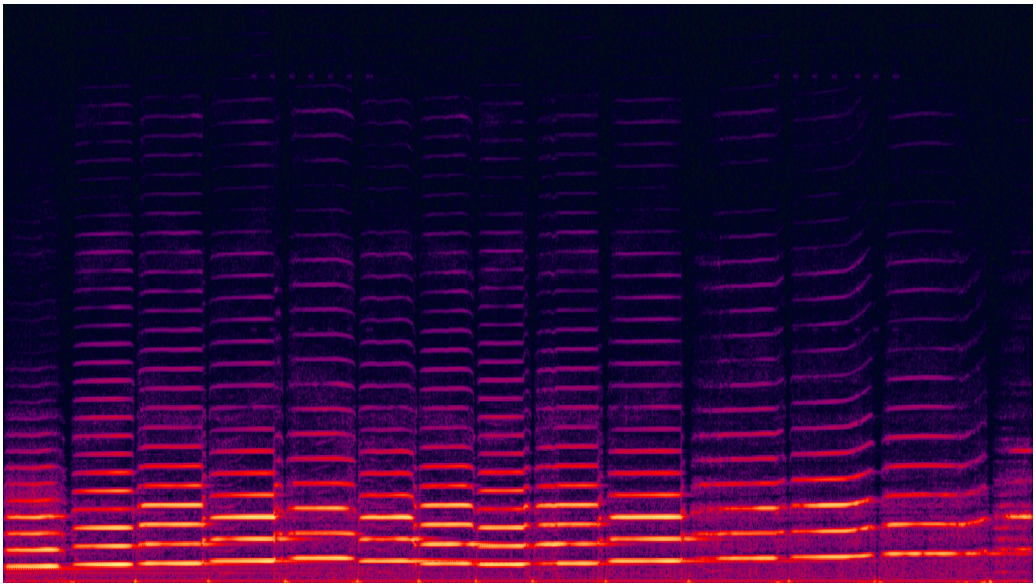

Figure 2: Spectogram

4.1.4 Time Frequency Resolution Tradeoff

Wideband vs Narrowband Spectogram [3]

Long window: narrowband spectogram

[3] Video:`https://www.youtube.com/watch?v=g1_wcbGUcDY`

- Long window → More DFT points → More frequency resolution
- Long window → More "things can happen" → less precision in time

Short window: wideband spectogram

- Short window → many time slices →precise location of transitions
- Short window → fewer DFT points → poor frequency resolution

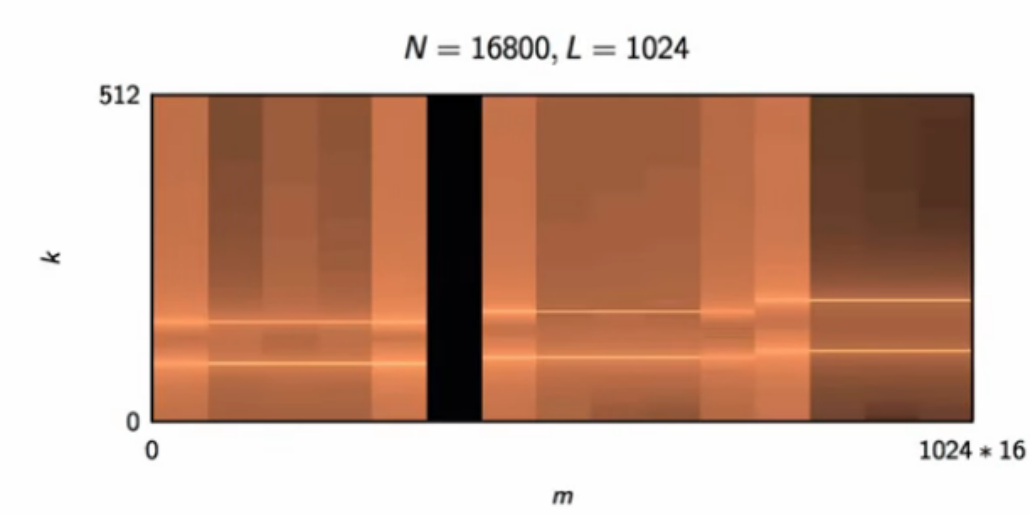

Figure 3: Narrow Band Spectogram

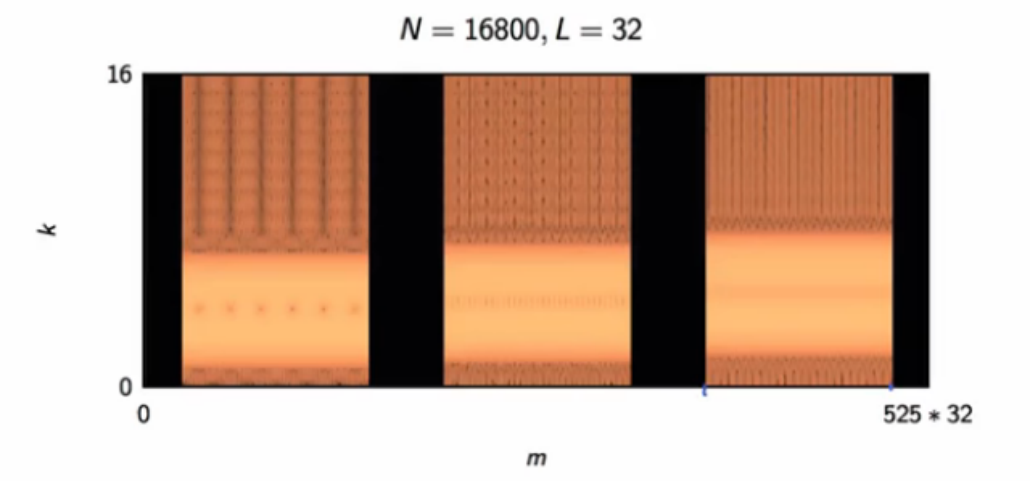

Figure 4: Wide Band Spectogram

4.2 Study of alignment algorithms

After extracting the features from audio and midi we get a multi-variate time sequence. We need to align these time sequences. A dynamic programming approach called Dynamic Time

Warping is used for this purpose. DTW [3] is an algorithm to search for an optimal alignment between two temporal sequences. It returns a distance measure for gauging similarities between them. Sequences are allowed to have local non-linear distortions in the time dimension, and DTW handles local warpings to some extent.

4.2.1 Goal of Dynamic Time Warping

A multi-variate time series $\mathcal{T}$ is a sequence of real valued vectors,i.e $\mathcal{T} = (t_1, t_2, \ldots, t_L)^T$. Given 2 sequences $\mathcal{P}$ and $\mathcal{Q}$ of possible different lengths $\mathcal{L}_\mathcal{P}$ and $\mathcal{L}_\mathcal{Q}$ namely $\mathcal{P} = (p_1, p_2, \ldots, p_{\mathcal{L}_\mathcal{P}})^T$ and $\mathcal{Q} = \left(q_1, q_2, \ldots, q_{\mathcal{L}_Q}\right)^T$ and let $\mathcal{D}(\mathcal{P}, \mathcal{Q}) \in \mathcal{R}^{\mathcal{L}_\mathcal{P} \times \mathcal{L}_\mathcal{Q}}$ be an pairwise distance matrix between sequences $\mathcal{P}$ and $\mathcal{Q}$ where $\mathcal{D}(\mathcal{P}, \mathcal{Q})_{i,j}$ is the distance between p_i and q_i. Various distance measures can be used like Cosine[4] or Euclidean[5]. The goal of temporal alignment between P and Q is to find two sequences of indices α and β of the same length $l\,(l \geq \max(\mathcal{L}_\mathcal{P}, \mathcal{L}_\mathcal{Q}))$, which match index $\alpha(i)$ in the time series $\mathcal{P}$ to index $\beta(i)$ in the time series $\mathcal{Q}$ such that the total cost along the matching path $\sum_{i=1}^{l} \mathcal{D}(\mathcal{P}, \mathcal{Q})_{\alpha(i),\beta(i)}$ is minimized.The alignment path (α, β) is constrained to satisfies boundary, monotonicity and continuity conditions.

$$\begin{cases} \alpha(1) = \beta(1) = 1 \\ \alpha(l) = \mathcal{L}_\mathcal{P}, \beta(l) = \mathcal{L}_\mathcal{Q} \\ (\alpha(i+1), \beta(i+1)) - (\alpha(i), \beta(i)) \in \{(1,0), (1,1), (0,1)\} \end{cases}$$

Note that the third condition can be changed depending on the application. Various step schemes can be followed as depicted in the figure.

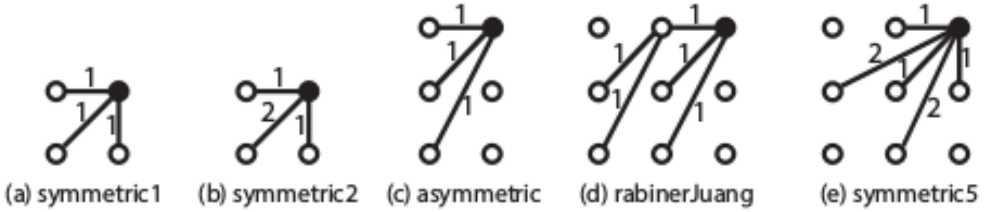

Figure 5: Step patterns

4.2.2 Description of the algorithm

To determine such an optimal alignment, one recursively computes a (PxQ)-matrix $\mathcal{M}$ where the matrix entry $\mathcal{M}(p, q)$ is the total similarity of the optimal alignment between $(x_1, \ldots, x_p)$ and $(y_1, \ldots, y_q)$ The choice of each step is recorded in a matrix ψ which is later used to compute an optimal path via backtracking.

$$M(p,q) := \min \begin{cases} M(p-1, q-1) + w_1 D(p,q), \psi[p,q] = 0(\nearrow) \\ M(p-1, q) + w_2 D(p,q), \psi[p,q] = 1(\rightarrow) \\ M(p, q-1) + w_3 D(p,q), \psi[p,q] = 2(\uparrow) \end{cases} \quad \text{for } p, q > 1.$$

[4] $1 - \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}}$

[5] $d(\mathbf{p}, \mathbf{q}) = d(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$

Furthermore, $M(p,1) := \sum_{k=1}^{p} w_2 D(k,1)$ for $p > 1$, $M(1,q) = \sum_{k=1}^{q} w_3 D(1,k)$ for $q > 1$. and $M(1,1) := D(1,1)$

The weights $(w_1, w_2, w_3) \in R_+^3$ can be used to adjust the preference over the three step sizes.

However DTW can become computationally expensive for longer time sequences. Hence some constraints like Sakoe–Chiba and Itakura bands are applied that are widely used to constrain the search area; limiting the search area tremendously aids the improvement of the speed. Some variants of DTW and their comparison is discussed in Section 6.

4.3 Study of evaluation procedure

The MIREX forum mentions the evaluation procedure for audio-score alignment. For evaluation we need ground-truth alignment between audio and midi.

4.3.1 Datasets for evaluation

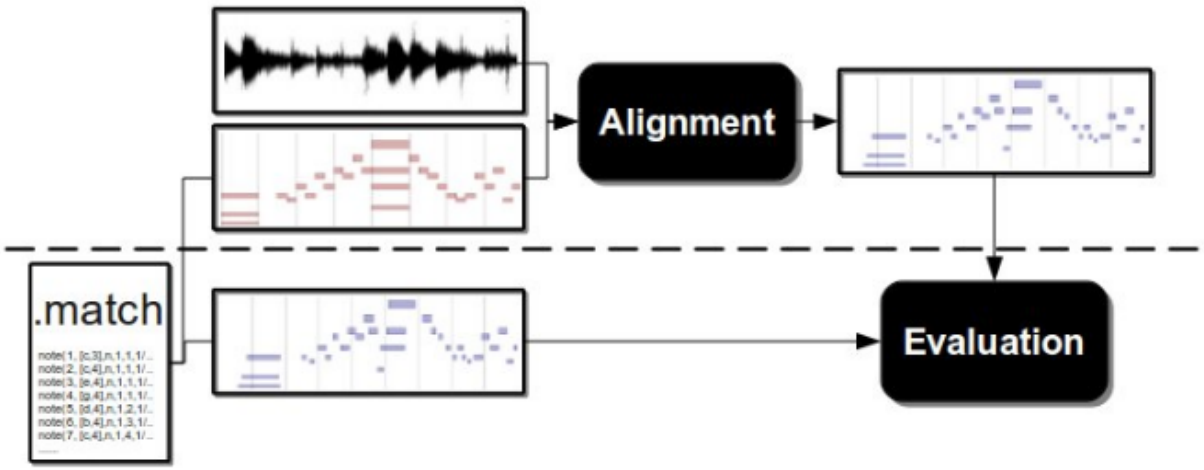

Figure 6: Evaluation framework

In this work we have used the Bach10 dataset [4] for evaluation purposes. Moreover we have provided a modified version of the Bach10 dataset which is discussed in Section 7. Bach10 is a polyphonic music dataset which can be used for versatile research problems, such as Multi-pitch Estimation and Tracking, Audio-score Alignment, Source Separation, etc. This dataset consists of the audio recordings of each part and the ensemble of ten pieces of four-part J.S. Bach chorales, as well as their MIDI scores, the ground-truth alignment between the audio and the score, the ground-truth pitch values of each part and the ground-truth notes of each piece. The audio recordings of the four parts (Soprano, Alto, Tenor and Bass) of each piece are performed by violin, clarinet, saxophone and bassoon, respectively. This work uses 3 files for each song from the collection of 10 songs.

- .wav: Performance audio of the ensemble of the piece.
- .mid: The MIDI score of the piece

- .txt: the modified MIREX format of the ground-truth alignment between audio and MIDI. It is a data array of four columns. Each row corresponds to a MIDI note. From left to right, columns correspond to:
  - onset time in audio (in ms)
  - onset time in MIDI (in ms)
  - pitch in MIDI (in MIDI number)
  - channel number of this note (violin=1, clarinet=2, saxophone=3, bassoon=4)

  We are concerned only with the first 2 columns.

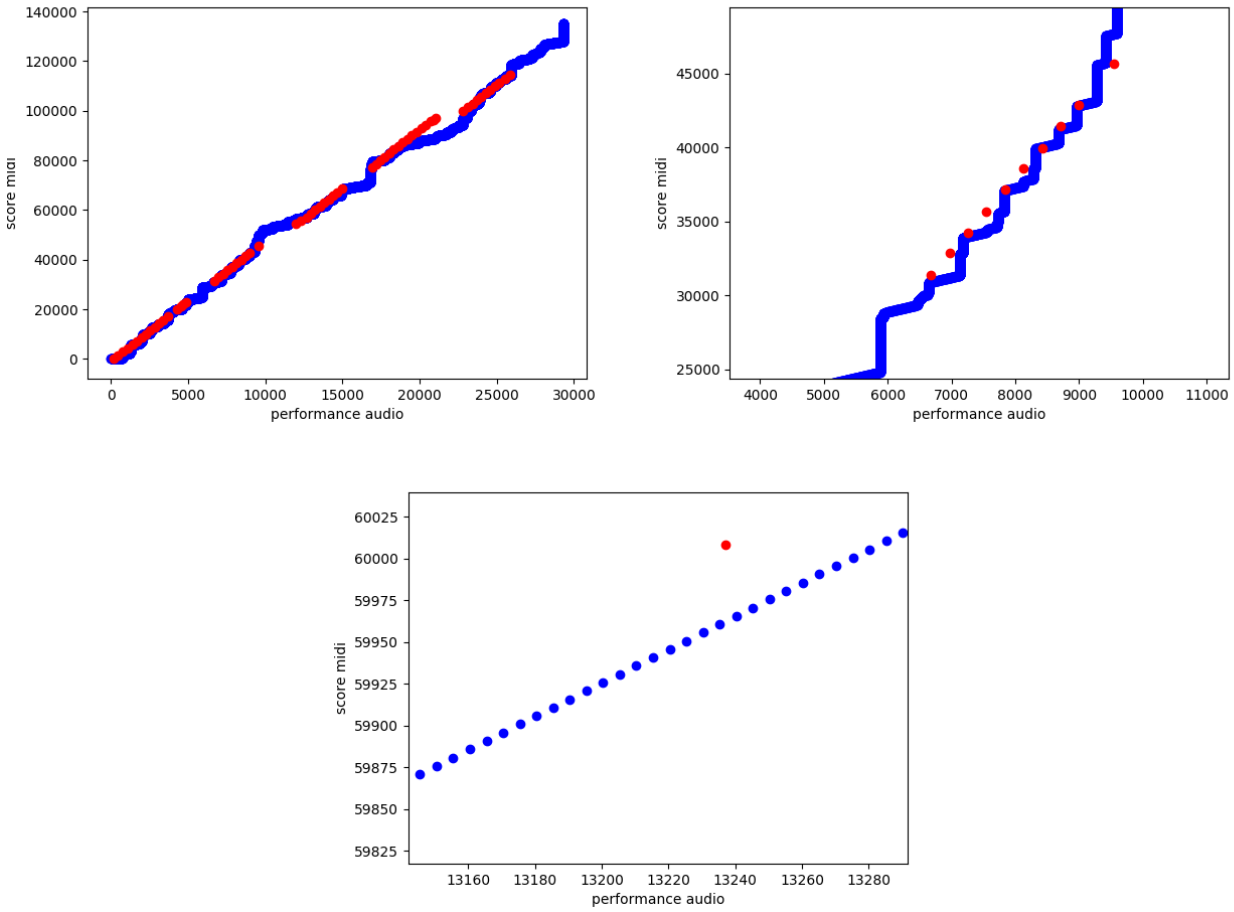

Figure 7: Ground truth alignment(red) and Alignment from our system(blue).
Fig 7 b) and c) are the zoomed in versions of Fig 7 a)

4.3.2 Evaluation Framework

As per Figure 6, the region above the dotted line is visible to the alignment system. The input to the alignment system is the performance audio file and the midi score file. The output is the alignment file (framewise alignment in this work as obtained from DTW system). The input

to the evaluation system is the ground truth alignment file between audio and midi(obtained from the Bach10 or Modified Bach10 dataset) and the file generated from the alignment system. Note that the file generated from DTW is a frame wise alignment and the file from ground truth alignment is a note wise alignment as mentioned in 4.3.1. Figure 7 is a graphical comparison of the 2 files.

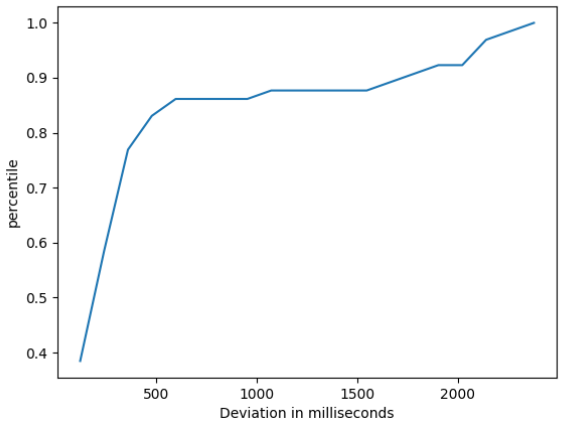

Figure 8: Cumulative distribution plot of the deviation

4.4 Review of neural networks relevant in MIR

This section discusses some neural networks which will be used in some models. RNN and LSTM are used in the experimental work for comparison of audio features described in section 7.1 and the auto-encoders for extracting the transposition invariant features [5] discussed in section 5.

4.4.1 RNN and LSTM

We have used RNN and LSTM in the Automatic Music Transcription system. The experiment is discussed in detail in section 5. Here we discuss how these networks can be used in the AMT tasks.

RNNs are favoured than feed forward neural networks for the AMT tasks due to restricted ability of feed forward neural networks for handling sequential data. RNNs are a better option for AMT applications as consecutive frames will include both present and past features.

$h^t_{layer+1} = f\left(W^f_{layer} h^t_{layer} + W^r_{layer} h^{t-1}_{layer} + b_{layer}\right)$ Where W^f_{layer} is the weight matrix of the forward pass of the layer W^r_{layer} is the weight matrix for the recurrent connection and b_{layer} the Bias vector. The parameters W^f_{layer}, W^r_{layer} and b_{layer} are estimated using using the back propagation through time algorithm (BPTT) and SGD.

LSTM are a kind of RNNs architecture capable of learning long term dependencies by using the memory cell. The update step is done over 4 neural networks included in each memory cell commonly called gates. The stored value or state vector is not iteratively squashed over time, and the gradient does not tend to vanish during Back-propagation. Due to its main characteristics, LSTMs have replaced traditional RNNs for the majority of sequential/time series tasks.

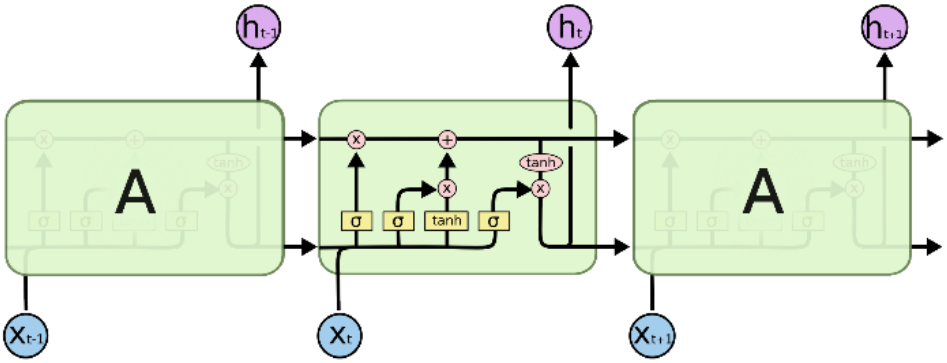

Figure 9: Long Short Term Memory Networks

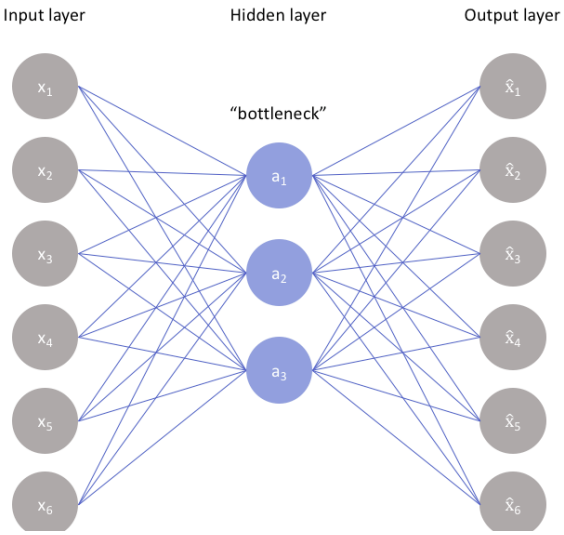

Figure 10: Autoencoder Network

4.4.2 Autoencoders

Autoencoders are an unsupervised learning technique in which we leverage neural networks for the task of representation learning. The aim is to design a neural network architecture which imposes a bottleneck[6] in the network which forces a compressed knowledge representation of the original input. If the input features were each independent of one another, this compression and subsequent reconstruction would be a very difficult task. However, if some sort of structure exists in the data (i e. correlations between input features), this structure can be learned and consequently leveraged when forcing the input through the network's bottleneck. The method is to take an unlabeled dataset and frame it as a supervised learning problem tasked with outputting $\hat{x}$, a reconstruction of the original input x. This network can be trained by minimizing the reconstruction error, $\mathcal{L}(x, \hat{x})$, which measures the differences between our original input and the consequent reconstruction. Andreas Arzt and Stefan Lattner [5] have proposed a convolutional gated autoencoder architecture to extract the transposition invariant

[6]A neural network layer with less number of neurons then the layer below or above it

features from audio. We have used their architecture to extract these audio features and apply them for the alignment problem. The architecture and procedure for extraction of the transposition invariant features is discussed in detail in section 5.

5 Transposition Invariant Features

5.1 Motivation

As per the ISMIR 2018 paper [6], the inspiration behind using the notion of relative pitch is that it is common for people to perceive and memorize melodies in terms of pitch intervals (or in terms of contours, the upward or downward direction of pitch intervals) rather than sequences of absolute pitches. This characteristic of music perception also has ramifications for the perception of form in musical works, since it implies that transposition of some musical fragment along the pitch dimension (such that the relative distances between pitches remain the same) does not alter the perceived identity of the musical material, or at least establishes a sense of similarity between the original and the transposed material.

5.2 Model used for extracting these transposition invariant features

Goal of the model : Let $\mathbf{x}_t \in R^M$ be a vector representing the energy distributed over M frequency bands at time t. Given a temporal context $\mathbf{x}_{t-n}^t = \mathbf{x}_{t-n} \dots \mathbf{x}_t$ (i.e. the input) and the next time slice $\mathbf{x}_{t+1}$(i.e. the target), the goal is to learn a mapping $\mathbf{m}_t$ (i.e. the transposition-invariant feature vector at time t) which does not change when shifting $\mathbf{X}_{t-n}^{t+1}$ up- or downwards in the pitch dimension.

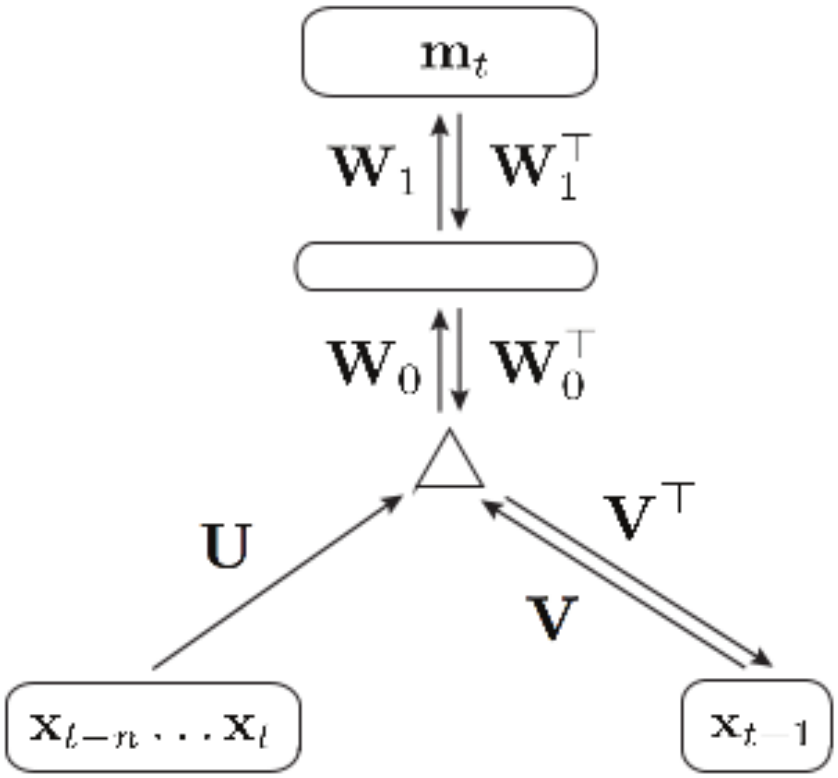

Figure 11: Convolutional Gated Auto Encoder

Architecture used : Gated Auto-Encoders (GAE) are used which utilize multiplicative

interactions to learn correlations between or within data instances, in this case to learn representations that represent the relation between the music at some point t and the preceding musical context.

Parameters used in the model :

- Context length,i.e width of input window: 9
- N-bins(number of frequency bins for CQT): 120
- No of output channels after the first convolution: 256
- No of output channels after the 1st mapping layer:128
- No of output channels after the 2nd mapping layer:64
- The shapes of 2 kernels U and V mentioned in Fig 11 are (9,120) and (1,120) respectively where U is applied on the input frame and V on the target frame.

The mapping at time t is calculated as $\mathbf{m}_t = \sigma_h\left(\mathbf{W}_1\sigma_h\left(\mathbf{W}_0\left(\mathbf{U}\mathbf{x}_{t-n}^t \cdot \mathbf{V}\mathbf{x}_{t+1}\right)\right)\right)$ where $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}_k$ are weight matrices, and σ_h is the hyperbolic tangent non-linearity. The operator $\cdot$ (depicted as a triangle in Figure 11) denotes the Hadamard (or element- wise) product of the filter responses $\mathbf{U}\mathbf{x}_{t-n}^t$ and $\mathbf{V}\mathbf{x}_{t+1}$ denoted as factors. The target of the GAE can be reconstructed as a function of the input $\mathbf{x}_{t-n}^t$ and a mapping $\mathbf{m}_t$: $\tilde{\mathbf{x}}_{t+1} = \mathbf{V}^\top\left(\mathbf{W}_0^\top\mathbf{W}_1^\top\mathbf{m}_t \cdot \mathbf{U}\mathbf{x}_{t-n}^t\right)$

Details of training : We have trained the model on the songs chosen from the MAPS dataset(around 20 randomly chosen piano songs with average of 4 min recording per .wav file). The CQT for each .wav file is calculated(CQT because of the experimental results from Section 7.1). The CQT obtained is separated into magnitude and phase parts and only the magnitude part is used. These calculations are done from Librosa library functions. For training, the frame is divided into blocks of 1024 and fed to the network in batches of 5. We have trained for 200 epochs with the loss function (MSE loss function) MSE $= \frac{1}{M}\left\|\mathbf{x}_{t+1} - \tilde{\mathbf{x}}_{t+1}\right\|^2$. A novel feature of training as mentioned in the paper to train the network to learn the transposition invariant features is to shift the value of the frequency dimension in the input randomly in the range (-60,60).

Extraction of the transposition invariant features: After the training is complete, we can convert any .wav file to get the required transposition invariant features. The .wav file is input to the network and the mapping obtained without transposing the input and target vectors is stored in a matrix. This mapping is a representation of the transposition invariant features which can be used in the place of standard chroma features for alignment as is used in the conventional methods.

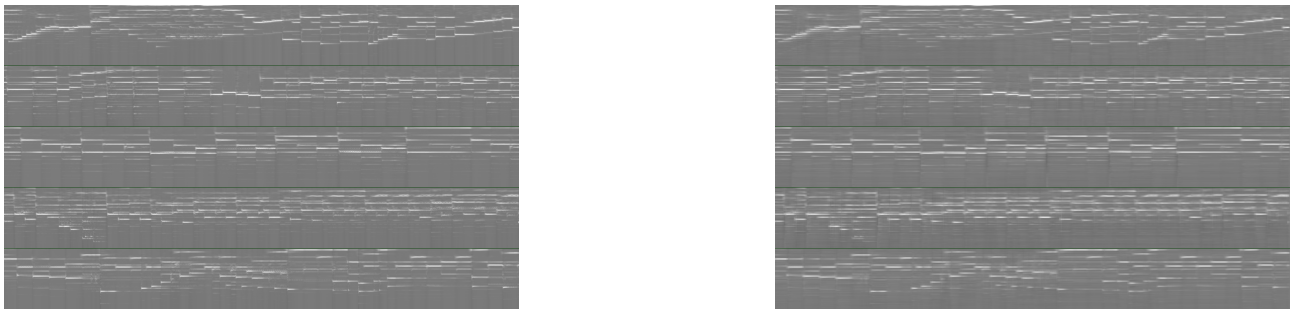

Figure 12: Original input and reconstruction

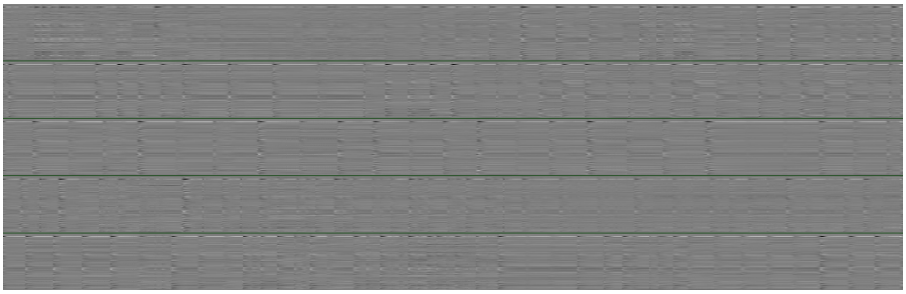

Figure 13: Mapping

6 Variants of DTW

In this section we will focus on 2 aspects of DTW. Section 6.1 deals with reducing the time complexity of DTW [7]. Section 6.2 deals with incorporating local features instead of point to point alignment.

6.1 FastDTW

6.1.1 Motivation

Standard dynamic time warping (DTW) is an $O(N^2)$ algorithm because every cell in the cost matrix must be filled to ensure an optimal answer is found, and the size of the matrix grows quadratically with the size of the time series. The quadratic time and space complexity of DTW creates the need for methods to speed up dynamic time warping. The methods used to make DTW faster can be categorized as:

- Constraints – Limit the number of cells that are evaluated in the cost matrix.
- Data Abstraction – Perform DTW on a reduced representation of the data.
- Indexing – Use lower bounding functions to reduce the number of times DTW must be run during time series classification or clustering.

6.1.2 Procedure for computation of FastDTW

- Coarsening – Shrink a time series into a smaller time series that represents the same curve as accurately as possible with fewer data points.
- Projection – Find a minimum-distance warp path at a lower resolution, and use that warp path as an initial guess for a higher resolution's minimum-distance warp path.
- Refinement – Refine the warp path projected from a lower resolution through local adjustments of the warp path.

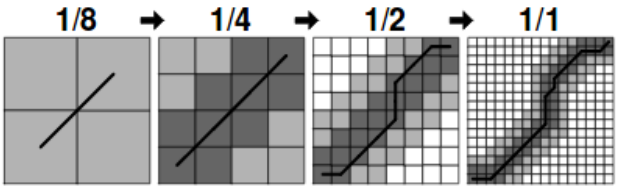

Figure 14: FastDTW

6.2 ShapeDTW

6.2.1 Motivation

DTW finds a global optimal alignment under certain constraints, but it does not necessarily achieve locally sensible matchings. Jiaping Zhao [8] proposed a method which incorporates local shape information around each point into the dynamic programming matching process, resulting in more semantically meaningful alignment results, i.e., points with similar local shapes tend to be matched while those with dissimilar neighborhoods are unlikely to be matched.

6.2.2 Procedure for computation of ShapeDTW

ShapeDTW consists of two steps:

- Represent each temporal point by some shape descriptor
- Align two sequences of descriptors by DTW or FastDTW.

Given a univariate time series $\mathcal{T} = (t_1, t_2, \ldots, t_L)^T, \mathcal{T} \in \mathcal{R}^L$ shapeDTW begins by representing each temporal point t_i by a shape descriptor $d_i \in \mathcal{R}^m$ which encodes structural information of temporal neighborhoods around t_i in this way, the original real value sequence $\mathcal{T} = (t_1, t_2, \ldots, t_L)^T$ is converted to a sequence of shape descriptors of the same length, i.e, $\mathrm{d} = (d_1, d_2, \ldots, d_L)^T, \mathrm{d} \in \mathcal{R}^{L \times m}$ shapeDTW then aligns the transformed multivariate descriptor sequences d by DTW, and at last the alignment path between descriptor sequences is transferred to the original univariate time series sequences

6.2.3 Shape descriptors

Shape descriptor	Description
Raw-Subsequence	$d_i = \mathcal{I}(s_i) = s_i$ where $\mathcal{I}(\cdot)$ is the identity function.
Piecewise aggregate approximation (PAA)	The mean value of temporal points falling within each interval is calculated and a vector of these mean values gives the approximation of Si , $\mathcal{F}(\cdot) = PAA, d_i = PAA(s_i)$
Discrete Wavelet Transform (DWT)	$\mathcal{F}(\cdot) = DWT, d_i = DWT(s_i)$

7 Experiments and observations

7.1 Comparison of audio features

For finding the best audio features, we compared the accuracy obtained in an AMT system with different features as input to the network. As the aim of this experiment was to compare the different audio features and not on the transcription accuracy, we used a generic network consisting of LSTM cells. The features chosen for comparison were:

- Chroma STFT
- Chroma CQT
- MFCC

The features were extracted from the audio using the functions from the Librosa Library.

The order of accuracy obtained on these features by using the same architecture is .

$CQT > STFT > MFCC$

So CQT was chosen as the feature for further experiments as it reported good accuracy compared to the other features.

7.2 Chroma STFT vs Transposition Invariant Features

This experiment deals with the comparison of 2 features. The Bach10 dataset is used for the evaluation. In each of the experiments, the comparison is done between the 2 audio files: One is the performance audio which is available as .wav file and the other audio file is synthesized using MuseScore software from the MIDI files of the ensemble of the pieces. The features are then extracted from these audio files and compared using FastDTW with radius[7] as 50. We calculate the absolute deviation at the aligned reference points as was discussed in Section 4.3.2. The 1st quartile, median and 3rd quartile absolute deviation over the 10 songs of the Bach10 has been graphically represented in Fig17, Fig18, Fig19 respectively. The Y axis represents the absolute deviation in milliseconds for each of the 3 figures.

7.3 Alteration of Midi files and ground truth

We used Pretty Midi library to alter the original MIDI files by mapping the original note onset timings to new onset timings which were obtained by randomly adding offsets and changing the tempo. As discussed in Fig 6, when we alter the midi files we also need to alter the ground truth files for the evaluation framework. The original MIDI and ground truth text files as well as the altered MIDI and ground truth text files have been publicly made available.[8] One can also refer this [9] for further modifications. We have provided only the significant results in this report. One can refer this [10] for future updates of this project.

[7] Increasing the radius increases the accuracy as well as the computation costs, so there is a tradeoff

[8] `https://drive.google.com/open?id=1RuGryTOyeS8H_hGC10Z68HKUfXk-K2Mz`

[9] `https://github.com/craffel/alignment-search/blob/master/corrupt_midi.py`

[10] `https://drive.google.com/open?id=1snPTOGBY50HshlAjOOJ_Ol2JjIiShA3s`

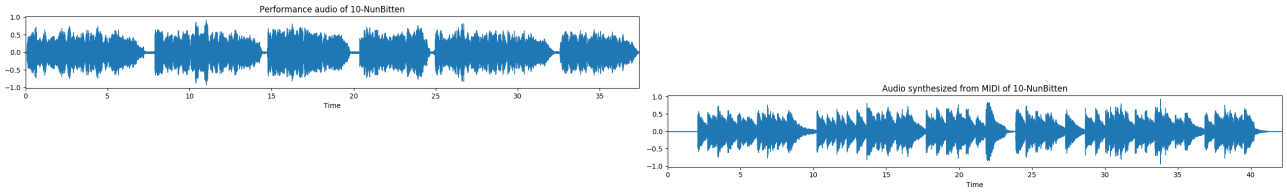

Figure 15: Performance audio and audio synthesized from MIDI for 10-NunBitten song

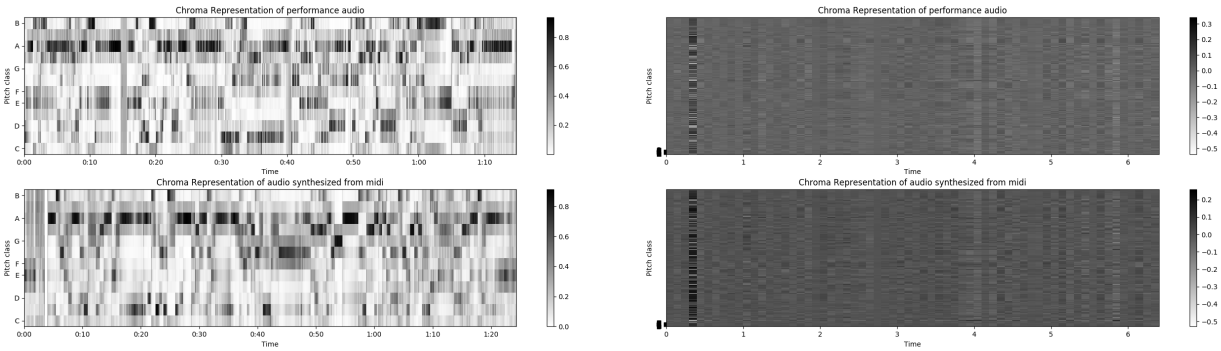

Figure 16: Chroma stft vs Transposition Invariant features for 10-NunBitten song

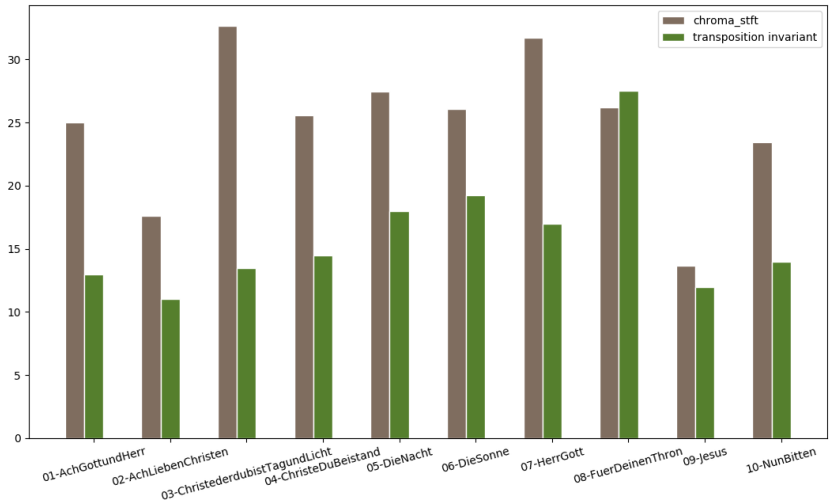

Figure 17: 1st Quartile deviation

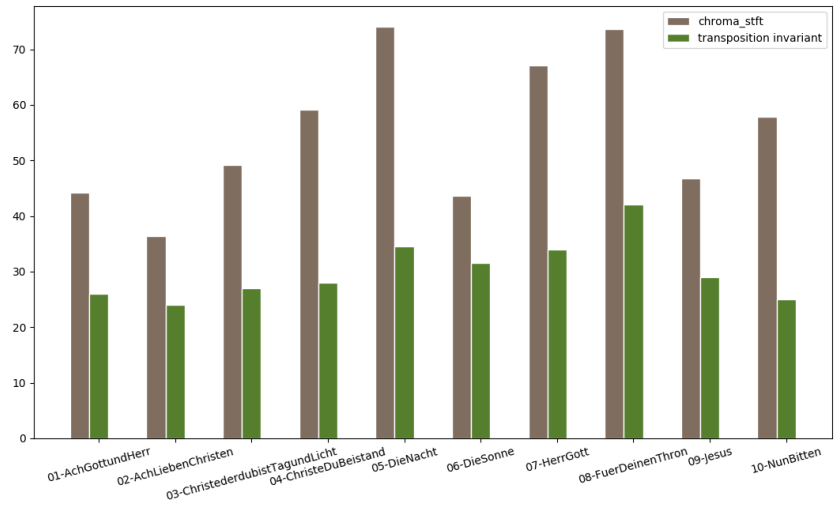

Figure 18: Median deviation

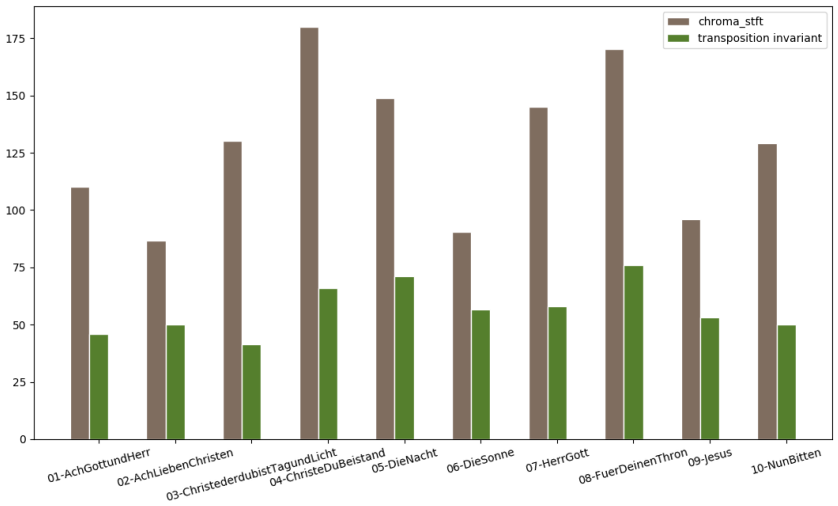

Figure 19: Third Quartile Deviation

8 Conclusions and scope of future work

We draw the following conclusions based on our observations:

- The transpositional invariant features obtained from the convolutional gated autoencoder are far better than the standard chroma features.
- ShapeDTW takes into account the local temporal changes.
- FastDTW reduces the time complexity of the standard DTW algorithm.

Scope of Future Work

- This alignment system form a base for the applications like music tutoring system. So for we have developed an evaluation scheme for calculating the error of the alignment algorithm. However the problem of calculating the performer's error which is a part of the music tutoring system is a bit different because we need to inform the performer the inserted, deleted and substituted notes.
- Incorporate this alignment scheme in Real time audio-score alignment systems.

References

[1] Kosuke Suzuki et al. "Real-time audio to score alignment using locally-constrained dynamic time warping of chromagrams". In: (2011).

[2] Benjamin Blankertz. "The constant Q transform". In: *URL http://doc. ml. tu-berlin. de/bbci/material/publications/Bla_constQ. pdf* (2001).

[3] Meinard Müller. "Dynamic time warping". In: *Information retrieval for music and motion* (2007), pp. 69–84.

[4] Z Duan and B Pardo. *Bach10 dataset*. 2015.

[5] Andreas Arzt and Stefan Lattner. "Audio-to-score alignment using transposition-invariant features". In: *arXiv preprint arXiv:1807.07278* (2018).

[6] Stefan Lattner, Maarten Grachten, and Gerhard Widmer. "Learning transposition-invariant interval features from symbolic music and audio". In: *arXiv preprint arXiv:1806.08236* (2018).

[7] Stan Salvador and Philip Chan. "Toward accurate dynamic time warping in linear time and space". In: *Intelligent Data Analysis* 11.5 (2007), pp. 561–580.

[8] Jiaping Zhao and Laurent Itti. "Shapedtw: shape dynamic time warping". In: *Pattern Recognition* 74 (2018), pp. 171–184.