A benchmark and evaluation of Motif set definitions on musical compositions and lyrics

Bachelor thesis exposé

Jörn-Hagen Stoll
Humboldt University of Berlin

October 18, 2022

Contents

1 Introduction 1

2 Background 2

3 Related work 4
3.1 Motiflets – fast and accurate detection of Motifs in time series [11] 4

4 Objectives 5
4.1 Data collection 5
4.2 Data organization 5
4.3 Data preparation 5
4.4 Benchmarking and presentation 6

5 Results 6

6 Sources 7

1 Introduction

A time series (Figure 1) is a sequence of real-valued data points which are typically ordered by time. [3] Whereby a Motif (Figure 2) is a frequent subsequence of a time series T, which is approximately repeating in T. If a pattern is approximately repeating, this means that there has to be an underlying reason in the process of data generation for the pattern to not degrade, also referred to as the pattern is *conserved*. In common use cases, this might be the muscle activation when a bird flaps its wings or the rhythm of a musical composition. Motif detection is an unsupervised learning problem, meaning that no labelled datasets are required to execute the detection, making it especially useful for exploratory data analysis. Yet, there is a lack of labelled ground truth data to evaluate the results of Motif discovery algorithms. Other than the two mentioned concrete use cases, Motif detection has been used in a wide variety of fields, because of its versatility, such as medicine [1] and biology [2]. In the scope of this thesis, the first ground truth annotated benchmark using lyric-annotated musical compositions will be created and evaluated using existing state-of-the-art Motif discovery algorithms. Those annotations can be organized using *lrc* files [12], a file type which annotates lyrics to audio files, whereby a time interval is set to a verse by setting the initial time in centiseconds where the verse should be displayed. The advantage of this process is that Motifs in audio recordings can be for the first time identified as words, phrases, or verses of the lyric, making it possible to easily interpret them. This makes the validation and reason why and how those Motifs are formed widely accessible. In 1990 Vanilla Ice allegedly committed a copyright infringement in his song "Ice Ice Baby" by not crediting Queen and Bowie in their Song "Under Pressure". The similarities lie in their bass line, which is what is investigated in the stumpy documentation [25]. In Figure 1 both musical compositions are plotted using their MFCCs (Mel-frequency cepstral coefficients) [13]. To detect Motifs across compositions, both are concatenated such that a new time series T of length $m_a + m_b$, with time series of the first composition starting at Index 0, ending at $m_a - 1$ and time series of the second composition starting at Index m_a, ending at $m_a + m_b - 1$ is created. From the concatenated time series, motif discovery can be executed, which then returns the kth best Motif pair. The Motifs from both compositions are plotted in Figure 3. [25] After manual visual verification, the similarity of the two musical compositions is at the found Motif pair indexes. Using this example, the detection of Motif sets across musical compositions is possible, but the problem faced is that in order to form a fully automated benchmark on a supplied algorithm, a ground truth has to be delivered. Using lyrical data corresponding to a musical composition as ground-truth, frequent lyric patterns can be analyzed, based on which the motif detection can be parameterized. After evaluating the results of each motif detection algorithm, found motifs can be verified and compared, after which the performance of a motif algorithm can be assessed.

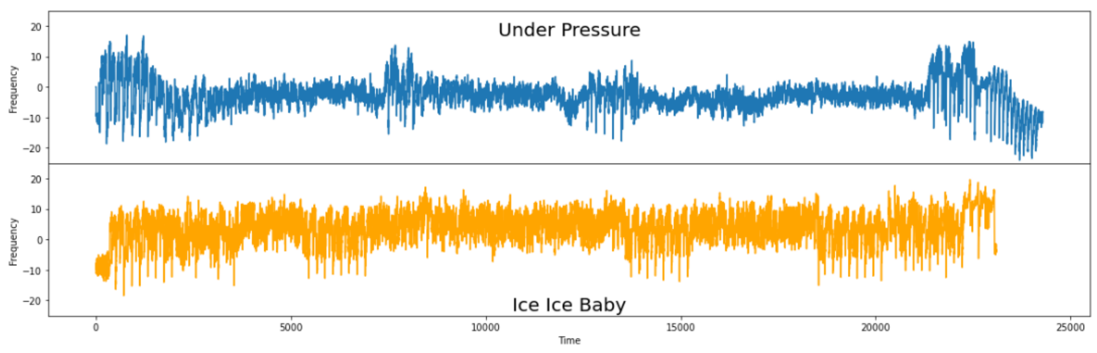

Figure 1: Two time series visualizations were created by plotting the Frequency of the musical compositions "Ice Ice Baby" and "Under Pressure" sampled at 100 Hz using a single MFCC channel. [25]

2 Background

In the literature regarding Motif discovery, various definitions exist, which is why the formal definitions in the context of this project are specified as followed.

Definition 2.1. A time series $T = (t_1, \ldots, t_n)$ is an ordered sequence of n real-valued variables $t_i \in \mathbb{R}$.

For analyzing time series, a crucial first step is to partition it into smaller components.

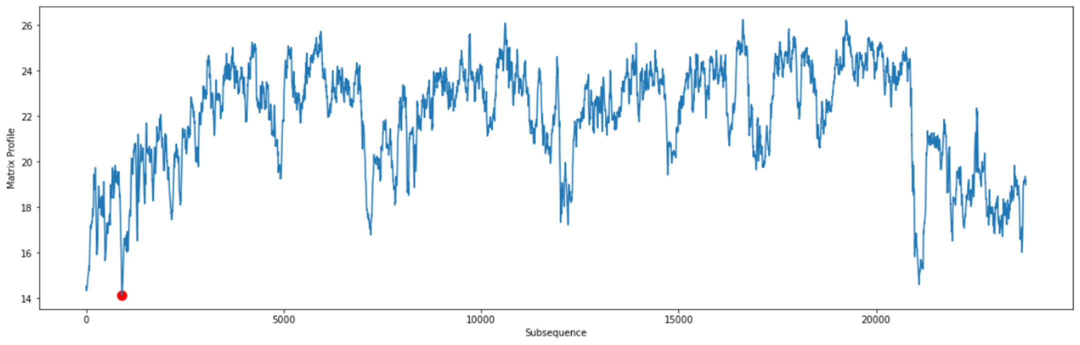

Figure 2: The computed Matrix profile and its global minimum (red dot) [25]

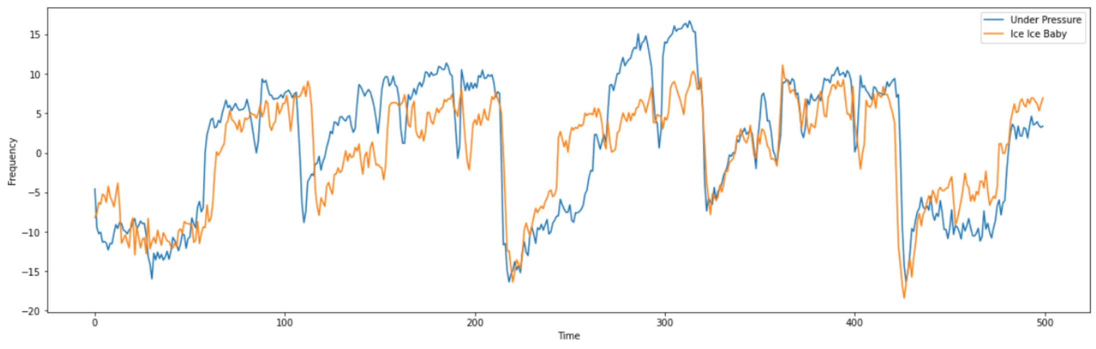

Figure 3: Visualization of the top Motif pair from both compositions [25]

Definition 2.2. Subsequence: Given a time series $T = (t_1, ..., t_n)$ of length n, C_p is a subsequence of T starting at offset p and length $m < n$ if $C_p = (t_p, \ldots, t_{p+m-1})$ for $1 \leq p \leq n-m-1$ [3]

One of the most trivial ways to compare two subsequences is to compare their distance.

Definition 2.3. Overlapping subsequences: A trivial match or overlap occurs when two subsequences A_i and B_j with length m in a time series T if and only if (or iff) the subsequences share not less than $m/2$ similar offsets in T : $(i - m/2) \leq j \leq (i + m/2)$ [11]

Given two subsequences that do not overlap, next a distance function must be established to evaluate the similarity of the match.

Definition 2.4. z-ED: The *z-normalized Euclidean distance* measures the distance between two series A and B of length n. Using the means μ_A and μ_B, standard-deviations σ_A and σ_B of A and B. z-ED is calculated using the following formula:

$$z - ED(A,B) = \sqrt{\sum_{i=1}^{n} \left(\frac{a_i - \mu_A}{\sigma_A} - \frac{b_i - \mu_B}{\sigma_B} \right)^2} \quad (1)$$

[11]

By using the distance function and setting a threshold, a set of similar subsequences can be found.

Definition 2.5. r-Match: Given a distance function D, a threshold $r \in \mathbb{R}$ and a time series $T = (t_1, ..., t_n)$, the subsequence C_p r-matches or matches (if not stated otherwise) the subsequence M_q iff $z - ED(C_p, M_q) \leq r$ and C_p, M_q are non-overlapping. [3]

Other than finding a set of r-matches of uncertain size, a common use case is the evaluation of two subsequences that have minimal distance from each other.

Definition 2.6. Top Motif pair: A Motif pair of length m of a time series T are the two non-overlapping subsequences of T that have minimal distance among all pairs of subsequences of T. [11]

When analyzing a Motif, another way to find the top Motif is to determine the number of r-Matches of the subsequence in the time series for a given input radius r.

Definition 2.7. Top K-Motif: Given a time series T of length n and a threshold $r \in \mathbb{R}$, the 1-Motif is the subsequence C_q of length m with the highest amount of r-matches, while the 2-Motif is the subsequence C_q with the second-highest amount of r-matches. [11]

In order to define k-Motiflets, the extent of a set of subsequences must be defined.

Definition 2.8. Extent: Given a time series T and a set of subsequences $S = (S_1, S_2, ...)$, each of length m. The *extent* of the subsequences in S is the maximal pairwise distance from the subsequences of S:

$$extent(S) = max_{(S_1, S_2 \in S \times S)}(z - ED(S_1, S_2))$$

[11]

Next, using the extent, the k-Motiflet can be defined. A difficulty when trying to explore Motif sets using the previous definition of K-Motif sets is setting the right threshold r. K-Motiflets solve that problem by only parameterizing the Motif set size to describe the largest set of k similar subsequences.

Definition 2.9. Top k-Motiflet: For a time series T with length n and a cardinality $k \in \mathbb{N}$ the top k-Motiflet is the Set S where the cardinality of S is k, meaning $|S| = k$, and the following holds: All subsequences in the set S are pairwise *r-matching* and no set S' exists such that $extent(S') < extent(S)$. [11]

3 Related work

3.1 Motiflets – fast and accurate detection of Motifs in time series [11]

This paper introduces the Motiflets and showcases the detection of k-Motiflets using the algorithm published in this GitHub repository [10] using ECG, muscle activation, sleep spindles and winding data.

4 Objectives

A difficulty when comparing Motif set algorithms is the lack of available labelled ground truth data, which is why this project's goal will be to acquire ground-truth labelled datasets to be benchmarked to detect the Motif sets and evaluate the results in the context of musical compositions, especially sound data and their lyrical annotations. Acquiring those annotations has not been done yet in the context of Motif detection and in a broader context, there is little annotated data at all, whereby audio data is available with their corresponding annotations in the form of "lrc" files, which allows finding Motif sets and compare them to the ground-truth derived from the lyrics of musical compositions. Generally, the evaluation will be pursued in the scope of this project for musical data.

What needs to be considered are the kind of Motif sets which can be found using lyrical data as ground-truth. For this use case, one premise is the similarity of the lyrics, meaning that Motif sets in melodies without any alphabetic similarity will not be able to be verified, although they can be found by a motif algorithm. Other than that, a high similarity in the spelling of a word does not always correspond to a high similarity in terms of pronunciation, which is something that cannot be taken into consideration when working with lyrical data. For instance, the words "tomb", pronounced "/tu:m/", needs one manipulation of the letter "t" to "b" to reach the word "bomb", pronounced "/bom/", but the difference in terms of pronunciation is significantly larger. The fact that Motif set detection will be evaluated on musical compositions also comes with the issue that words are frequently vocalized differently to make the word fit into the rhythm or to give a higher or lower emphasis on a word, this is for instance the case for the Queen & Bowie composition "Under pressure" [17] in the timestamp 1m07s with the word "higher" with a high emphasis and in timestamp 1m04s with "pray" as a low emphasis. Using lyrics as a basis to detect Motifs harbours another issue because two musical compositions with very similar lyrical annotations might not share a common melody, making the detection of common motifs harder to accomplish.

Motif sets will be detected using a selection of Motif discovery algorithms.
The tasks can be broken down into the following four steps:

4.1 Data collection

First, musical compositions must be found where their corresponding lyrical annotations are available. A good start and proof-of-concept is the motif detection based on an "a cappella" composition or an "a cappella" version of a composition. This eliminates any instrumental influence on motif detection. As an example, the "A cappella" version of "Under Pressure" by Queen and Bowie [18] can be used followed by its original version [17].

4.2 Data organization

Since audio files are usually compressed into audio formats like "mp3" [20], the file has to be uncompressed and organized into a "csv" file. If a compressed audio file is present, it can be uncompressed using "ffmpeg" [20] into a "wav" [22] file, which can be read using Scipy [23]. Further, the corresponding "lrc" has to be read, but that task is trivial since "lrc" files are encoded in plain text.

4.3 Data preparation

In terms of preparation, lrc data has to be parsed into a Pandas data frame as a time series and the related audio data will be parsed into frequency coefficients using MFCC [13] with the support of "python_speech_features" [19]. For parsing "lrc" files, "pylrc" [24] can be used, which avoids writing a custom parser. Also, by parsing lyrical data into a prefix tree, frequent words and verses can be extracted. The definition of the “lrc” file format states [20], that a word is initiated using a tag <mm:ss.xx> e.g. <00:17:25> followed by the word itself and a verse is initiated using a tag with the format [mm:ss:xx] e.g. [00:12:25], followed by a sequence of words initiated by their corresponding tags and ended with a line break “\n” or the end of string notation “\0”. Using this notation, the set of words W and the set of verses V in a given lyrical dataset can be extracted, based on which the prefix tree can be built.

4.4 Benchmarking and presentation

By using Motif discovery on the algorithms EMMA [14], k-Motiflets [10], Latent Motifs [15] and Set Finder [16] the benchmark will be evaluated using the mentioned algorithms. Using the prefix tree, the selected algorithms can be parameterized and executed by extracting common patterns in the lyrical data, based on which the radius r, the motif set size k and the length m are set from the audios melody. Afterwards, the results will be evaluated using Pandas [6] and the plotting libraries Seaborn [9] and Matplotlib [8]. A valid result is found if the Motif based on audio data overlaps with the query derived from the prefix tree using lyrical data.

5 Results

The primary programming language used will be Python [4] using Jupyter Notebooks[5]. The Python libraries used are Pandas [6] to manage Dataframes, Stumpy [7] to calculate matrix profiles (Figure 3) and find patterns and Motifs, as well as the plotting libraries Matplotlib [8] and Seaborn [9]. The results will be documented within the Jupyter Notebook and made public to a GitHub repository to supply further detailed information beyond the scope of the paper.

6 Sources

[1] Tian, B., &amp; Mathews, M. B. (1970, January 1). Phylogenetics and functions of the double-stranded RNA-binding motif: A genomic survey. Rutgers, The State University of New Jersey. Retrieved August 2, 2022, from https://www.researchwithrutgers.com/en/publications/phylogenetics-and-functions-of-the-double-stranded-rna-binding-mo

[2] Abou&nbsp;Assi, H., Garavís, M., González, C., &amp; Damha, M. J. (2018, August 16). I-motif DNA: Structural features and significance to Cell Biology. OUP Academic. Retrieved August 2, 2022, from https://academic.oup.com/nar/article/46/16/8038/5075032

[3] Finding time series motifs in disk-resident data. IEEE Xplore. (n.d.). Retrieved August 2, 2022, from https://ieeexplore.ieee.org/abstract/document/5360262

[4] Welcome to Python.org. Python.org. (n.d.). Retrieved August 2, 2022, from https://www.python.org/

[5] Project jupyter. Project Jupyter. (n.d.). Retrieved August 2, 2022, from https://jupyter.org/

[6] Pandas. pandas. (n.d.). Retrieved August 2, 2022, from https://pandas.pydata.org/

[7] TDAmeritrade. (n.d.). TDAmeritrade/stumpy: Stumpy is a powerful and scalable Python Library for Modern Time Series analysis. GitHub. Retrieved August 2, 2022, from https://github.com/TDAmeritrade/stumpy

[8] Visualization with python. Matplotlib. (n.d.). Retrieved August 2, 2022, from https://matplotlib.org/

[9] Statistical Data Visualization. seaborn. (n.d.). Retrieved August 2, 2022, from https://seaborn.pydata.org/

[10] Patrickzib. (n.d.). Patrickzib/motiflets: Motiflets. GitHub. Retrieved August 2, 2022, from https://github.com/patrickzib/motiflets

[11] Schäfer, P., &amp; Leser, U. (2022, June 8). Motiflets – fast and accurate detection of motifs in time series. arXiv.org. Retrieved August 2, 2022, from https://arxiv.org/abs/2206.03735v1

[12] .LRC file&nbsp;extension. LRC File Extension - What is an .lrc file and how do I open it? (2017, November 9). Retrieved August 8, 2022, from https://fileinfo.com/extension/lrc

[13] Zheng, F., Zhang, G., & Song, Z. (2001). Comparison of different implementations of MFCC. Journal of Computer science and Technology, 16(6), 582-589.

[14] Lonardi, J. L. E. K. S., & Patel, P. (2002, August). Finding motifs in time series. In Proc. of the 2nd Workshop on Temporal Data Mining (pp. 53-68).

[15] Grabocka, J., Schilling, N., & Schmidt-Thieme, L. (2016). Latent time-series motifs. ACM Transactions on Knowledge Discovery from Data (TKDD), 11(1), 1-20.

[16] Bagnall, A., Hills, J., & Lines, J. (2014). Finding motif sets in time series. arXiv preprint arXiv:1407.3685.

[17] (2010, March 26). Queen &amp; David Bowie, under pressure - lyrics. YouTube. Retrieved September 13, 2022, from https://www.youtube.com/watch?v=zG1G5abvMsU

[18] MisteRhapsody. (2011, July 20). Under pressure (A-capella) - only vocals. YouTube. Retrieved September 13, 2022, from https://www.youtube.com/watch?v=uMQb9LCNGxs

[19] Welcome to python_speech_features's documentation!¶. Welcome to python_speech_features's documentation! - python_speech_features 0.1.0 documentation. (n.d.). Retrieved August 16, 2022, from https://python-speech-features.readthedocs.io/en/latest/

[20] .mp3 file&nbsp;extension. MP3 File Extension - What is an .mp3 file and how do I open it? (2022, March 22). Retrieved August 16, 2022, from https://fileinfo.com/extension/mp3

[21] FFmpeg. (n.d.). Retrieved August 16, 2022, from https://ffmpeg.org/

[22] .WAV file&nbsp;extension. WAV File Extension - What is a .wav file and how do I open it? (2022, February 24). Retrieved August 16, 2022, from https://fileinfo.com/extension/wav

[23] Input and output (scipy.io). Input and output (scipy.io) - SciPy v1.9.0 Manual. (n.d.). Retrieved August 16, 2022, from https://docs.scipy.org/doc/scipy/reference/io.html?highlight=wav#module-scipy.io.wavfile

[24] Pylrc. PyPI. (n.d.). Retrieved August 16, 2022, from https://pypi.org/project/pylrc/

[25] TDAmeritrade. (n.d.). Stumpy/tutorial_ab_joins.ipynb at main · tdameritrade/stumpy. GitHub. Retrieved August 16, 2022, from https://github.com/TDAmeritrade/stumpy/blob/main/docs/Tutorial_AB_Joins.ipynb