Identify musics using data compression (NCD)

Lab Work 3

Bruno Nunes | 80614
Catarina Marques | 81382
David Raposo | 93395

Introduction

The ability to automatically identify music is a reality in modern times. One approach to music identification is based on the concept of Normalized Compression Distance (NCD), a metric from algorithmic information theory. NCD measures the similarity between two objects based on their compressed representations, making it a powerful tool for identifying similarities in complex data such as audio files.

This project aims to develop and test a system for automatic music identification using NCD. The fundamental idea is to use standard compression algorithms to compare music segments with a database of complete tracks. By transforming audio files into a representation suitable for general-purpose compressors and computing the NCD values between the query segment and the tracks in the database, the system can identify the most similar music.

To do this, we will employ several standard compression algorithms, including *gzip*, *bzip2*, *lzma*, and *zstd*. The effectiveness of these compressors in the context of music identification will be evaluated by comparing their performance on a dataset. Additionally, we will introduce noise to the query segments to test the robustness of the NCD-based identification technique.

The process starts by converting the audio files into a more compressor-friendly format, extracting the most significant frequencies from overlapping segments of the audio. These frequency representations will then be compressed and used to calculate NCD values, allowing the closest match in the database to be identified. Comparative analysis of various compressors will provide insights into their effectiveness for this task.

Methodology and Algorithm

In order to attend to the objective of testing a Normalized Compression Distance (NCD) setup for automatic identification of musics, a python program (taizam.py) was developed to process a dataset of musics into a form of signature, used to compute the NCD from compression with a smaller sample of a specified music. In this section are presented the program developed, the dataset used and the procedure of experiments to evaluate this strategy for music identification.

The program (taizam.py)

The program developed has 2 main functionalities: the signatures database generation and the music identification. In order to experiment with the robustness of the strategies, this program accepts noise and sample trimming factors and others as input parameters.

Database functionality

Transforming the music files in their frequency domain, also referred as a kind of signature, is the suggestion given in the assignment to improve the success chance of the music identification process. This means that for every comparison made, we need to have the original music's and sample's signature file.

With the intent of improving overall performance we decided to not introduce in music identification functionality the generation of the signature file for the dataset original music being queried. For this reason, we developed a signature database generation process, which basically consists in computing each dataset music's signature file and storing it in a defined directory. On the repository of this project this is located under the "Data" directory, on "Signatures" folder.

Music identification functionality

Music identification is the core functionality of this project. When selected this function through the input parameters, the program starts by extracting a sample from the original music selected as requested. Then noise is added to this sample and the final product is the signature file for this sample. This procedure implies the generation of three different files obeying to the following structure:

- Sample file:
    - {music name}_s{sample start time}_d{sample duration}.wav
- sample file with noise mixed in:
    - {music name}_s{sample start time}_d{sample duration}_n{noise level}.wav
- Signature file for the sample:
    - No noise: {music name}_s{sample start time}_d{sample duration}.wav.sig
    - With noise: {music name}_s{sample start time}_d{sample duration}_n{noise level}.wav.sig

Each of these files are stored also under the directory "Data", on "temp" folder", as they are considered temporary files that are not needed after the identification process.

The need of generating these files is a consequence of using third-party command line/terminal tools to perform the segmentation, noise mixing and signature generation.

Sampling, noise mixing and signature generation

For segmentation of the sample and noise mixing functions, SOX[1] is used as suggested in the assignment. A class called "AudioProcessor" was created in songs_handling.py to interact with this tool. This class was designed with methods to explore audio properties like

[1] http://sox.sourceforge.net/

its duration, sampling rate and the number of channels, and methods to segment according to a defined duration and a start time, and to add noise within a defined period of time and volume level.

Regarding the audio signature file generation, the GetMaxFreqs tool is used at the given state. From this tool the only need is to perform the signature computation. To do so, a method was created to request this signature files for .wav music files exclusively. This method is the same used for the signature database generation and the sample signature generation.

Classification

In the process of inferring at which music belongs the processed sample, its signature file is used. The NCD is calculated through compressions of the sample's signature file and a original music signature file, separately (C(x) and C(y)) and concatenated(C(x,y)) in accordance to the following formula:

$$NCD(x,y) = \frac{C(x,y) - min\{C(x),C(y)\}}{max\{C(x),C(y)\}}$$

The program allows the usage of four different compression algorithms: gzip, bzip2, lzma, zstd.

Distance calculation is performed with every single signature file available on the signature database directory ("Data/Signatures/"), classifying as the original music the one to which the distance is shorter.

Input parameters

To allow the described operation and manipulation, the as the following arguments:

- **<Process>** - task to perform [0 – signature database generation, 1 – music classification]
- **<Compressor>** - compression algorithm [gzip, bzip2, lzma, zstd]
- **<wavFile>** - the original music file name (full path)
- **--sampleStart <sampleStart>** - percentage of time at which the sample starts
- **--sampleDuration <sampleDuration>** - duration of the sample
- **--noiseLevel <noiseLevel>** - noise level in percentage [0 to 1]

An important thing to denote is the actual noise level. The input given to the program comes as a percentage of a maximum limit of 50% noise level. While testing the generated samples with noise we noticed that from this level on the sample files were mainly noise. As so we considered them as irrelevant as they would certainly generate wrong inferences.

Datasets and Metrics

A music dataset was gathered for the purpose of the project. This data set is composed by in 35 music files in .wav format, where 7 are the samples provided for the assignment. The remaining different music tracks in .wav format downloaded from royalty-free sources like Youtube Audio Library, Incompetech[2] and Bensound[3].

This dataset is located on the repository of the project, in the directory of usage that is "Data/Database/"

As a way to measure the results, the computed NCD scores were collected for the original file where the sample tested is generated. Was also collected the lowest NCD score and the respective original file name associated as a representation of the identification result.

With these metrics accuracy was extrapolated for measuring the technique effectiveness.

Procedures

To evaluate the effectiveness of the Normalized Compression Distance (NCD) approach for automatic music identification, it is essential to test various combinations of parameters. The parameters considered include:

- Compressor Algorithms - compression algorithms used to compute the NCD (zlib, lzma, gzip, bz2).
- Sample Start Percentage - specifies the starting point of the music segment to be used.
- Sample Duration - indicates the length of the music segment in seconds.
- Noise Level - amount of noise added to the music segment.

The program was tested using different test cases to explore the program's behavior under various conditions:

1. Each compressor (*zlib*, *lzma*, *gzip*, *bz2*) with zero noise level (no noise added) and a duration of 10 seconds.
2. Set of tests using the two best compression algorithms with different noise levels (0, 0.1, 0.5, 1.0) and a specific sample configuration (duration 4s, 7s, 10s).

Results and Analysis

Accuracy test with no noise

The best NCD scores offer a quantifiable estimate of the test segments' closeness to the original music files. **Lower NCD scores indicate greater similarity.**

The bz2 compressor has the lowest average NCD score (0.945), followed closely by gzip (0.971). The zlib compressor has a significantly higher mean NCD score (0.973). The lzma compressor, which failed to detect several files, received an average NCD score of 0.362, indicating poor performance.

[2] incompetech.com/music/royalty-free/music.html
[3] bensound.com/free-music-for-videos

Accuraccy (samples with 10s and 0 noise)				
File	**bz2**	**gzip**	**lzma**	**zlib**
Adeste-Fideles-Shorter.wav	1.000	1.000	0.667	1.000
cozycoffeehouse.wav	1.000	1.000	0.000	1.000
sunlitdepths.wav	1.000	1.000	0.000	1.000
The_Throne_Silent_Partner.wav	1.000	1.000	0.000	1.000
Theme_for_Harold_var_3.wav	1.000	1.000	0.333	1.000

Table 1: Accuracy of identification of samples with 10s length and 0 noise.

Best NCD (samples with 10s and 0 noise)				
File	**bz2**	**gzip**	**lzma**	**zlib**
Adeste-Fideles-Shorter.wav	0.938	0.960	0.903	0.962
cozycoffeehouse.wav	0.962	0.985	--	0.986
sunlitdepths.wav	0.957	0.979	--	0.981
The_Throne_Silent_Partner.wav	0.948	0.972	--	0.974
Theme_for_Harold_var_3.wav	0.921	0.960	0.908	0.962
mean	0.945	0.971	0.905	0.973

Table 2: Best NCD computed for the correct identification of samples of 10s length and 0 noise.

From this data, considerations about the compressors at use can be taken:

- **Bz2**: This compressor had the best overall performance, with the highest accuracy and lowest average NCD score. This makes bz2 the most trustworthy alternative for NCD-based music identification.
- **Gzip** and **Zlib**: Both compressors performed admirably, with flawless accuracy and comparable NCD scores to bz2. These compressors can be considered as viable options if necessary.
- **lzma**: The poor accuracy and NCD ratings suggest that lzma is ineffective for this application.

NCD Score vs Sample Duration vs Noise Level

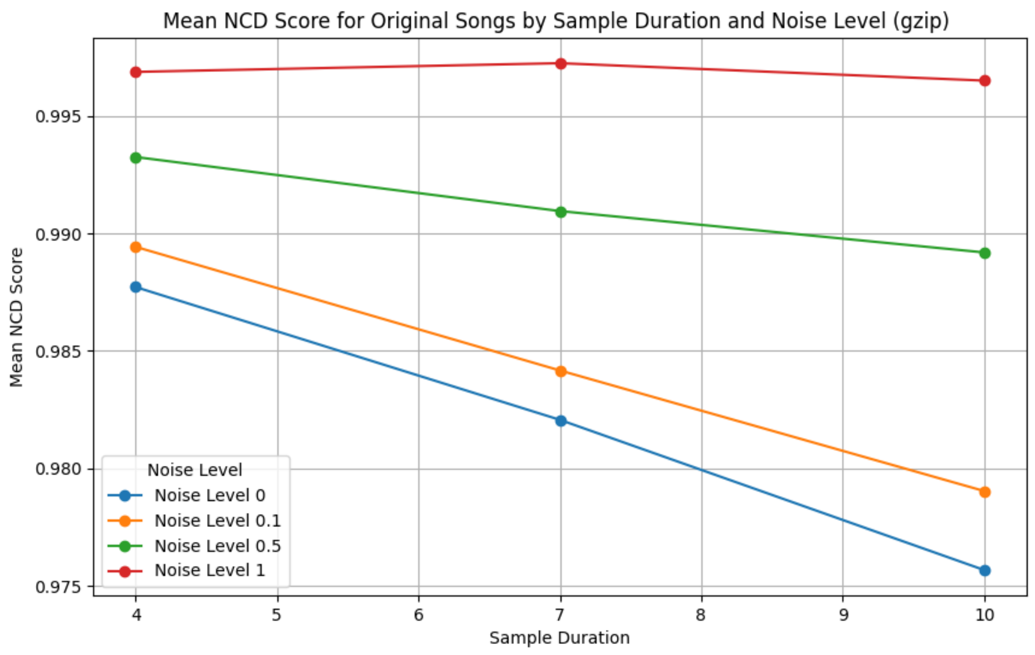

Figure 1: Graph of mean NCD computed with gzip between sample and original file in diferent sample noise and duration conditions.

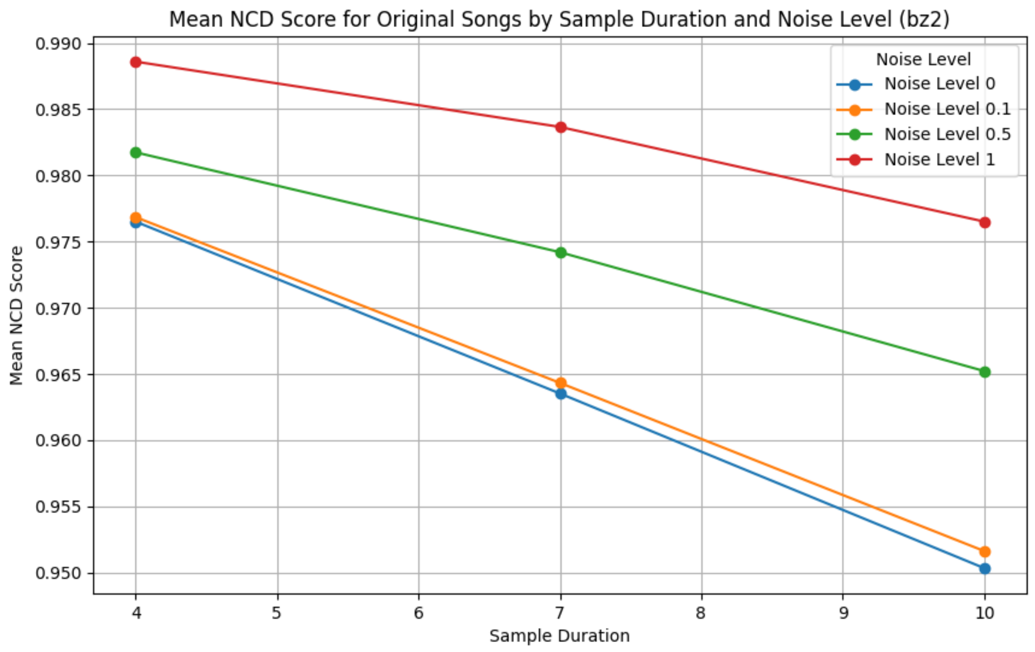

Figure 2: Graph of mean NCD computed with bz2 between sample and original file in diferent sample noise and duration conditions.

As it can be seen in the graphs above, longer sample durations enhance the compressor's ability to identify similarities, reflected in lower NCD scores.

Noise adversely affects the NCD score, but increasing the sample duration helps to counteract this effect.

Global Accuracy

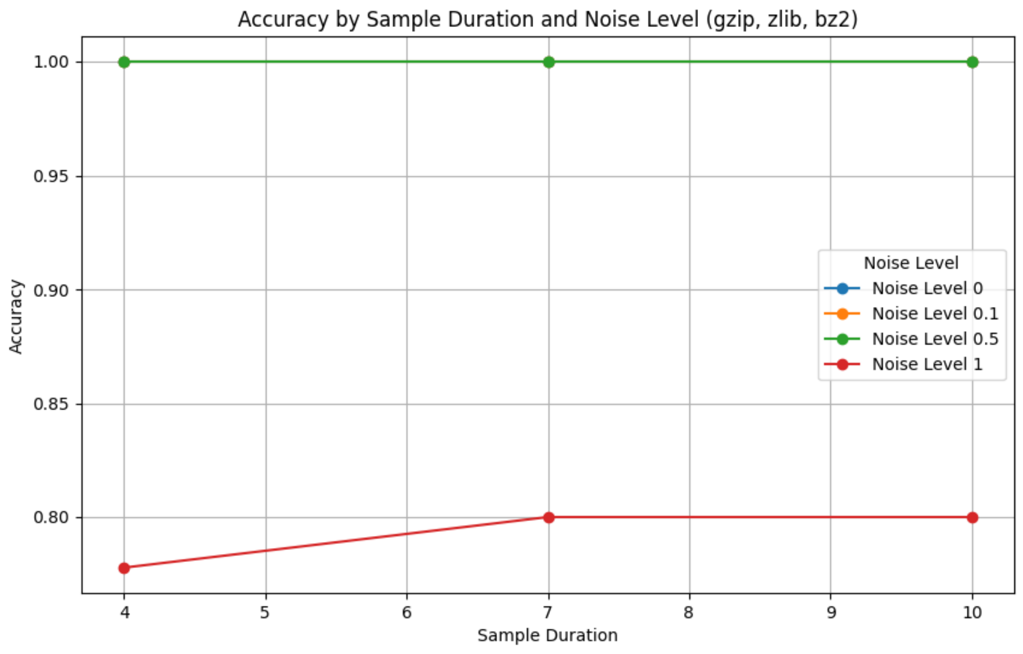

Figure 3: Global accuracy.

The results here presented reflect the total number of correct identifications among the different test parameter combinations (sample starting point, sample duration and noise level). From this we can read the following:

- For noise levels 0, 0.1, and 0.5, the accuracy remains at 100% across all sample durations.
- At the highest noise level (1.0), accuracy drops to approximately 78-80% for all sample durations.

These results indicate that the NCD technique, particularly with the bz2 and gzip compressors, is well-suited for music identification, offering high accuracy and robustness even in the presence of noise.

Conclusion

With the work to which this report refers to, we can access different aspects about the possibility of application of NCD technique in the field of music identification.

Through the compression with no noise we can conclude that for “clear” music, the sample identification by NCD can be almost guaranteed as the accuracy near 100% indicates. An important consideration here can be the compression algorithm applied to the technique. In our specific case bz2, gzip and zlib algorithms performed admirably and seem suitable for the use case while in the opposite side lzma points negatively.

Trying the identification of a damaged music sample, as is a sample contaminated by noise introduction, can also be accomplished with the applied methodology. For the different levels of noise applied we could observe that increasing noise level can increase the Normalized Compression Distance to the target, not interfering severely on the identification accuracy. As pinpointed on The Program section, the level of noise applied was limited to real maximum volume of 50% as it was considered by us reasonable. Above this level, the main context is noise and not music.

With a similar effect on NCD calculation was the sample size, in other words, the sample duration. Smaller samples make the task more difficult but not impossible. The NCD increases with the decrease of the sample information but despite it, the accuracy of the identification can remain high.

In the end we can confirm, within the universe tested in this assignment, that data compression can be used for the purpose of music identification through the Normalized Compression Distance (NCD) computation.

Future work

- Test with more compression algorithms
- Expanding the music database
- Test robustness with musics produced with similar audio samples as is happens nowadays
- Test other music file formats, different from .wav

Annexes

Step-by-Step Methodology

Step 1: Preparing the Dataset

1. **Search Music Files**: Download different music tracks in WAV format from royalty-free sources like Youtube Audio Library, Incompetech[4] and Bensound[5].
2. **Organize Files**: Store these music files in the "Database" directory.

Step 2: Sample handling

1. **Segment Files:** The selected file is segmented according to the parameters.
2. **Noise Addition:** If noiseLevel is specified, noise is added to the segment.

Step 3: Generating signatures

1. **Signature Generation:** Generate a signature file using the GetMaxFreqs.exe program.

Step 4: Calculating the Normalized Compression Distance (NCD)

1. **Compression:** Use the compression algorithm selected in the parameters (zlib, lzma, gzip, bz2) to compress the frequency representations.
2. **Compute NCD:** Use the formula to calculate de NCD

$$NCD(x,y) = \frac{C(x,y) - min\{C(x),C(y)\}}{max\{C(x),C(y)\}}$$

Step 5: Check the result

1. **Result:** Find the smallest distance to get the match.

[4] incompetech.com/music/royalty-free/music.html
[5] bensound.com/free-music-for-videos