Master thesis on Sound and Music Computing

Universitat Pompeu Fabra

Applying Audio Problem Detection Algorithms to Sounds on Freesound Web Platform

Ahmet Oğuz ÖZTÜRK

Supervisor: Dmitry BOGDANOV

Co-supervisor: Pablo ALONSO

July 2023

Master thesis on Sound and Music Computing

Universitat Pompeu Fabra

Applying Audio Problem Detection Algorithms to Sounds on Freesound Web Platform

Ahmet Oğuz ÖZTÜRK

Supervisor: Dmitry BOGDANOV

Co-supervisor: Pablo ALONSO

July 2023

Table of Contents

1. Introduction 1
1.1. Freesound 1
1.2. Audio Problems 1
1.3. Importance of Audio Problem Detection Algorithms 2
1.4. Objectives 3
1.5. Structure of the Study 3
2. State of the Art 4
2.1. Essentia 4
2.2. Detection Algorithms 4
2.3. Similar Studies 7
2.4. FSD50K Dataset 7
2.5. Differences Between Professional Audio Recordings and Sounds in FSD50K Dataset 9
3. Methodology and Results 12
3.1. Discontinuity Tests 12
3.2. Clipping Tests 16
3.3. Humming Tests 19
3.4. Gap Tests 22
3.5. Click and Pop Tests 27
3.6. Asymmetry Tests 28
4. Audio Analyzer in Freesound 31
5. Conclusions 32
6. List of Figures 34
7. List of Tables 35
8. Bibliography 36
9. Appendices 38

Abstract

With the domination of music broadcasting through the internet, detecting audio problems on sound files became more important. Essentia and other similar sound libraries enabled to detect audio problems automatically, which made the detection process faster, cheaper, and more accurate. These algorithms were mostly applied on music files which are usually recorded with adequate equipment in studio environment, and afterwards that are mixed and mastered. But there are many types of sounds other than music that can be recorded and uploaded to web. Freesound is a web platform that has currently the largest collection of sounds that are recorded in different environments, using different equipment, and tagged with suitable keywords. FSD50K is the dataset that includes 51,197 carefully annotated sounds gathered from Freesound Platform.

In this study, five of the mostly used automatic audio problem detection algorithms were selected. These are discontinuity, gap, click, hum, and clipping problems. Using Essentia, these algorithms were applied on FSD50K. Some bugs in the algorithms were detected and fixed, or some limitations were identified in their usage. For each audio problem, the sound classes that have the most problematic audio file percentages were detected.

Besides these already implemented algorithms, a detection algorithm for asymmetry, that is another property of sound, and which does not always have to be a problem, was implemented and results were analyzed.

An audio analyzer was implemented and added to Freesound Platform that allows users to filter the sound they are looking for based on whether the sound is problematic or not, according to the audio problem detection algorithms tested in the study.

Keywords: Essentia, audio problem detection algorithms, Freesound, FSD50K, discontinuity, hum, gap, click, clip, asymmetry.

1. Introduction

1.1. Freesound

Freesound is a popular web platform that provides a vast collection of user-generated audio files that can be used for various creative projects. It was founded in 2005 by the Music Technology Group at Pompeu Fabra University in Barcelona, Spain, and is now maintained by the Freesound team and community [1]. Platform offers a diverse range of sound effects, ambient sounds, and musical elements that can be incorporated into sound design projects that are mostly released under Creative Commons licenses that allow free usage.

Creators can search for specific sounds they need, such as footsteps, explosions, or animal sounds, and download them for use in films, video games, animations, or any other creative endeavors. Musicians and composers can find royalty-free music samples, loops, instrumentals, and individual sound elements. These audio files can be used as building blocks to create original compositions or enhance existing tracks. It allows artists to experiment with different sounds and textures, adding depth and uniqueness to their music. Podcasters and radio producers often require sound effects, background music, and other audio elements to enhance their productions. Game developers can leverage the platform to find sound effects and background music for their games. Filmmakers, video editors, and multimedia artists can utilize Freesound to enhance their audiovisual projects.

1.2. Audio Problems

Artists are becoming more interested in uploading their works on streaming services as these services overtake physical music stores as the primary source of music consumption. In many cases, artists are unable to upload their music without the help of a middleman who manages their contracts and royalties. Small and independent distributors cannot afford to develop their own technology to manage their catalogs, send releases to digital service providers (DSP), and collect royalties. Instead, music distribution companies offer this service.

An audio file goes through many stages until an artist records the sound and shares it on streaming sites. At these stages, errors may occur in the audio file for various reasons:

- Recording Quality: Low-quality audio recordings or inadequate microphone usage can result in unwanted sounds and low-frequency response.
- Noise and Background Interference: Electronic devices near the audio source or external factors (such as wind or traffic) can introduce noise or background interference that becomes audible in the audio files.
- Poor Conversion or Compression: When audio files are compressed or converted, data loss or quality degradation can occur.
- Input / Output Devices: Hardware or software misuse issues in the devices used for playing or recording audio files can cause the audio sources or output devices to malfunction.
- Encoding / Decoding Problems: The device or software intended for audio file usage may not support a specific audio format or fail to decode the file correctly.
- Interruptions or Faulty Operations: Breaks or errors that occur during the recording, copying, or transferring of audio files can corrupt the file and lead to audio problems.

1.3. Importance of Audio Problem Detection Algorithms

Music artists often use sound collections to enhance and enrich their music productions. Audio problems in sound collections can potentially carry over to the final music piece if they are not addressed properly. These problems can impact the overall quality and integrity of the music production.

Assuring a high standard in the quality of audio tracks uploaded to the system before sending them to the DSPs is one of the challenges faced by such distribution services in order to prevent any flaws in the distribution process. To achieve this, the services might rely on manual quality control (QC), which is time-consuming and expensive, because a QC agent must carefully listen to the tracks to ensure that there are no issues

with the audio. Instead of using manual quality control, many of the problems can be detected automatically using digital audio signal processing software.

1.4. Objectives

The audio problem detection algorithms are already used for professional audio recordings, but in this study, algorithms are applied to sounds in Freesound Platform, that are recorded in many different environmental conditions, and using many different recording equipment. The problems that occur during applying the algorithms to these sound files are investigated, reasons of the problems are identified and fixes or recommendations for the algorithms are proposed.

There are many different types of audio problems that can be detected automatically, but in this study discontinuity, clipping, click, gap and hum detection algorithms are tested. Also asymmetries of audio files are investigated; however asymmetry may not be identified as an audio problem in most of the cases.

1.5. Structure of the Study

In chapter 2, state of the art is explained, in chapter 3 methodology and results are given, chapter 4 includes audio analyzer implementation, and in chapter 5 conclusions take place.

2. State of the Art

2.1. Essentia

Essentia is an open-source software library and a set of tools for analysis, synthesis, and processing of audio and music signals, written in C++ and designed to be used in a variety of applications, including music information retrieval, audio classification, and sound synthesis [2]. Audio problem detection algorithms used in this study are already implemented in Essentia.

Usage of Essentia algorithms in music industry is first announced in a convention paper [3]. Validations have been done on a large music collection of more than 300,000 tracks.

2.2. Detection Algorithms

Essentia software includes many audio problem detection algorithms that have been implemented, tested, and currently in use. Details and input parameters of all Essentia functions can be found on the website [4]. For this study, five of these algorithms are selected, that are: discontinuity, clipping, hum, gap, and click detection algorithms. In addition to these algorithms, an algorithm to detect asymmetric waveforms is implemented.

2.2.1. Discontinuity

Discontinuity refers to a sudden or abrupt change in the waveform of the signal. This can occur when there is a sudden jump in the amplitude or phase of the signal, or when there is a sharp transition from one waveform to another. Discontinuities can cause a variety of unwanted effects, such as clicks, pops, and distortion. To detect discontinuity, Linear Prediction Coefficients (LPC) analysis [5] is carried out for each frame. Result of the LPC reconstruction is a predicted continuous frame. This predicted frame is subtracted from the original signal to calculate the prediction error. If the prediction

error exceeds an adaptive threshold, that is dependent on the standard deviation of the prediction error, then a discontinuity error is detected.

2.2.2. Clipping

Clipping is a type of distortion that occurs in audio signals when the amplitude of the signal exceeds the maximum level that the system can handle. When an audio signal is clipped, the peaks of the waveform are “clipped off” or truncated, resulting in a distorted signal with additional harmonics. To detect clipping, true peak detection algorithm [6] is used. The suggested algorithm operates by four times oversampling the input signal. An optional high shelf filter and a DC blocker are then available. Peaks above 0 dB on the absolute value of the resampled interpolated signal are regarded as clipped.

Interpolation converts the digital data to analog, after that detects if the analog signal exceeds the limits. So, it assures that it can find clipping even if samples do not have values at the limits, or it prevents false positives if one sample touch the limit but clipping does not occur.

2.2.3. Hum

Humming is a low-frequency noise that is typically caused by electrical interference in audio equipment or wiring. Hum can be heard as a constant, low-pitched tone that can be heard in the background of an audio signal, particularly when the audio signal is amplified or recorded at high levels. To detect hum, implementation proposed by Brandt and Bitzer [7] is used. Algorithm works by measuring the periodogram frequency bins steadiness in 10-30 seconds of audio segments.

2.2.4. Gap

Gap refers to a brief interruption in the waveform. This can occur for a variety of reasons, such as a dropout in the recording, a momentary loss of signal during

transmission, or intentional editing of the audio to remove unwanted noise or content. To detect gap, algorithm proposed by Mühlbauer [8] is used.

Gap detection algorithm distinguishes normal silences and gap errors using decay and attack parts of the audio. If an audio signal decays smoothly, and then a silence starts, then it is detected as a normal silence. But if an audio signal vanishes instantly, this part is detected as a gap error.

2.2.5. Click

Click refers to a sudden, transient sound that occurs as a result of an abrupt change in the audio waveform. Clicks are typically characterized by a brief, sharp, and noticeable spike in the audio signal. They can be caused by various factors, including technical issues during recording or editing, imperfections in the playback equipment or software, or errors in the encoding or decoding process of digital audio files.

Clicks in audio files can be undesirable because they can be distracting or disrupt the overall listening experience. They are particularly noticeable when the rest of the audio content is relatively quiet or when the click occurs during a pause or silence in the audio.

It is worth noting that, while clicks are often seen as audio imperfections, they can also be intentionally added for artistic purposes in certain genres or styles of music. For example, some electronic music genres incorporate clicks and glitches as part of their aesthetic. The proposed algorithm [9] was used to detect clicks in audio files.

In the article [10], deep networks were used to identify click problems. A huge dataset was used for training, and a small amount of improvement in click detection could be achieved.

2.2.6. Asymmetry

Asymmetry refers to a condition where the positive and negative halves of a waveform are not symmetrical or balanced around the centerline. It indicates that the shape,

amplitude, or timing of the positive and negative cycles of a signal is not identical or evenly distributed [11].

Asymmetry can be confused with DC offset. In DC offset problem, silences do not have zero amplitude, but have an offset on the positive or negative side. For asymmetry, silences have zero value, but the envelope shape of the signal tends to take place in one half of the axis, more than the other one, which leads to an asymmetry in overall shape. Audio asymmetry, itself, is not inherently an audio problem. It is a characteristic or quality that can be intentional or unintentional, depending on the artistic or technical intent behind the audio production. It results in an uneven dynamic range usage, so gain increase operations can lead to clipping problems.

By its definition, skewness can get both positive and negative values, so absolute value should be taken into account to compare two skewness values. In this study asymmetry is calculated as:

$$abs\ (\ skewness\ (\ audio\_sample\ )\ ) \tag{2.2.6}$$

where audio sample is output of MonoLoader function in Essentia, and skewness is the function from scipy library.

2.3. Similar Studies

Arteaga [12] studied on a detailed taxonomy of audio problems digitizing vinyl records. Badenas [13] applied similar Essentia algorithms on 1120 mono sounds from the test dataset for the Kaggle [14] and the most suitable parameters are investigated by comparing the result sets and ground truths. But because dataset is small and due to lack of time, study does not have satisfactory results.

2.4. FSD50K Dataset

The mentioned audio problem detection algorithms are currently being applied on music recordings, but in this study, algorithms are applied on a set of sounds from Freesound

platform. The dataset used in this study is FSD50K [15], which stands for “Freesound Dataset 50k”, which is a large-scale, open dataset of environmental sounds. The dataset contains 51,197 sound recordings, and were carefully annotated with a set of 200 labels that describe the sound’s content and context, drawn from the AudioSet Ontology [16]. These 200 labels are:

Accelerating and revving and vroom, Accordion, Acoustic guitar, Aircraft, Alarm, Animal, Applause, Bark, Bass drum, Bass guitar, Bathtub (filling or washing), Bell, Bicycle, Bicycle bell, Bird, Bird vocalization and bird call and bird song, Boat and Water vehicle, Boiling, Boom, Bowed string instrument, Brass instrument, Breathing, Burping and eructation, Bus, Buzz, Camera, Car, Car passing by, Cat, Chatter, Cheering, Chewing and mastication, Chicken and rooster, Child speech and kid speaking, Chime, Chink and clink, Chirp and tweet, Chuckle and chortle, Church bell, Clapping, Clock, Coin (dropping), Computer keyboard, Conversation, Cough, Cowbell, Crack, Crackle, Crash cymbal, Cricket, Crow, Crowd, Crumpling and crinkling, Crushing, Crying and sobbing, Cupboard open or close, Cutlery and silverware, Cymbal, Dishes and pots and pans, Dog, Domestic animals and pets, Domestic sounds and home sounds, Door, Doorbell, Drawer open or close, Drill, Drip, Drum, Drum kit, Electric guitar, Engine, Engine starting, Explosion, Fart, Female singing, Female speech and woman speaking, Fill (with liquid), Finger snapping, Fire, Fireworks, Fixed-wing aircraft and airplane, Fowl, Frog, Frying (food), Gasp, Giggle, Glass, Glockenspiel, Gong, Growling, Guitar, Gull and seagull, Gunshot and gunfire, Gurgling, Hammer, Hands, Harmonica, Harp, Hi-hat, Hiss, Human group actions, Human voice, Idling, Insect, Keyboard (musical), Keys jangling, Knock, Laughter, Liquid, Livestock and farm animals and working animals, Male singing, Male speech and man speaking, Mallet percussion, Marimba and xylophone, Mechanical fan, Mechanisms, Meow, Microwave oven, Motor vehicle (road), Motorcycle, Music, Musical instrument, Ocean, Organ, Packing tape and duct tape, Percussion, Piano, Plucked string instrument, Pour, Power tool, Printer, Purr, Race car and auto racing, Rail transport, Rain, Raindrop, Ratchet and pawl, Rattle, Rattle (instrument), Respiratory sounds, Ringtone, Run, Sawing, Scissors, Scratching (performance technique), Screaming, Screech, Shatter, Shout, Sigh, Singing, Sink (filling or washing), Siren, Skateboard, Slam, Sliding door, Snare drum, Sneeze, Speech, Speech synthesizer,

Splash and splatter, Squeak, Stream, Strum, Subway and metro and underground, Tabla, Tambourine, Tap, Tearing, Telephone, Thump and thud, Thunder, Thunderstorm, Tick, Tick-tock, Toilet flush, Tools, Traffic noise and roadway noise, Train, Trickle and dribble, Truck, Trumpet, Typewriter, Typing, Vehicle, Vehicle horn and car horn and honking, Walk and footsteps, Water, Water tap and faucet, Waves and surf, Whispering, Whoosh and swoosh and swish, Wild animals, Wind, Wind chime, Wind instrument and woodwind instrument, Wood, Writing, Yell, Zipper (clothing).

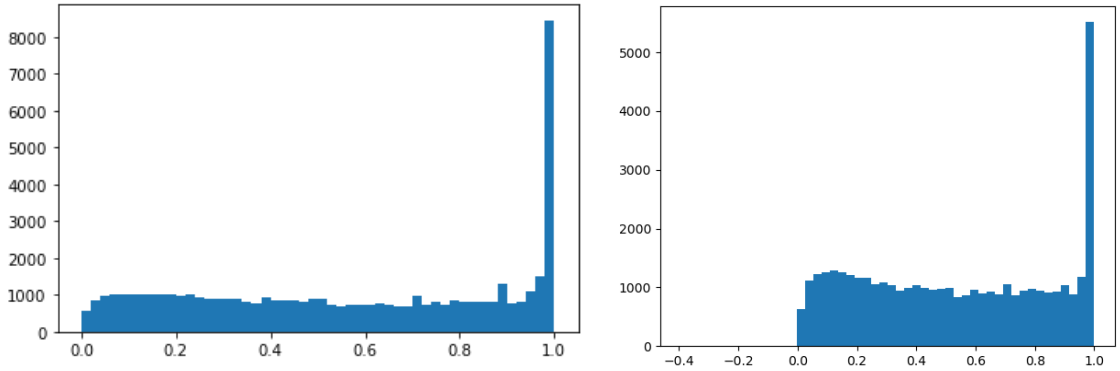
2.5. Differences Between Professional Audio Recordings and Sounds in FSD50K Dataset

Professional audio recordings and FSD50K dataset are two different types of audio sources with different characteristics and purposes.

Professional audio recordings are typically made in a controlled environment with high-quality equipment and experienced audio engineers. The goal is to capture the highest possible quality of sound, with a specific focus on the intended purpose of the recording, such as music production. Professional audio recordings are usually made with a specific artistic or commercial purpose in mind, and they are often processed, mixed, and mastered to achieve a specific sound or style. The mastering stage generally ensures that the errors at the signal level are eliminated or minimized.

On the other hand, FSD50K dataset is a collection of audio recordings gathered from a variety of sources, including amateur recordings, field recordings, and audio snippets contributed by users of the Freesound Platform. The recordings are diverse in terms of quality, background noise, and recording conditions, as they are not made under controlled conditions.

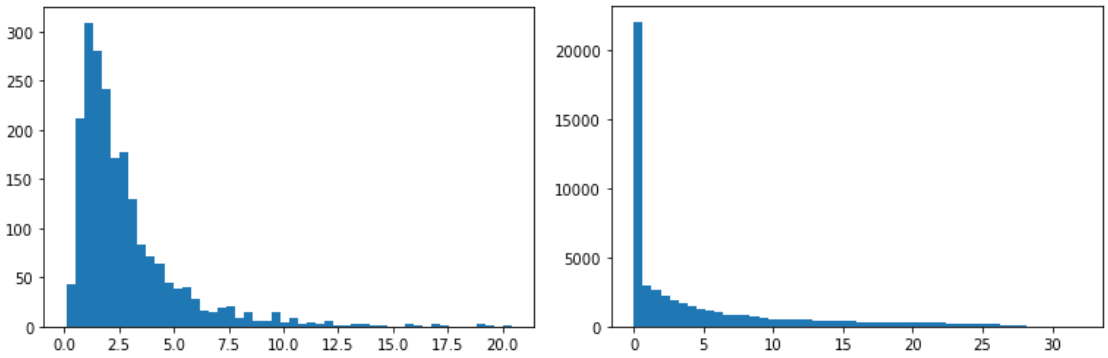
One of the most important outcomes of this difference can be seen in data ranges of the audio samples. For comparison, Mus-AV database [17] was used, which contains 2100 mastered music audio of the industry-level quality in mp3 file format. In Figure 1, comparison of histograms that includes maximum value of absolute values for each audio file is given.



(a) FSD50K dataset *(b) Mus-AV dataset*

Figure 1: Histogram comparison for max(abs(audio)) values calculated for each audio file.

In Figure 2, comparison of histograms that includes dynamic range values for each audio file is given. LoudnessEBUR128 method in Essentia library was used to calculate dynamic range of audio files.



(a) FSD50K dataset *(b) Mus-AV dataset*

Figure 2: Histogram comparison dynamic range values calculated for each audio file.

Another important difference is the length of the audio files. Bar chart of duration intervals of wav files in FSD50K dataset is given in Figure 3. There are 77 wav files that are longer than 30 seconds, and the longest wav file is 550 seconds long. 5943 of the files are less than 1 second, and the shortest of them is 300 milliseconds long. Duration histogram is important to analyze some of the audio problem detection algorithms.

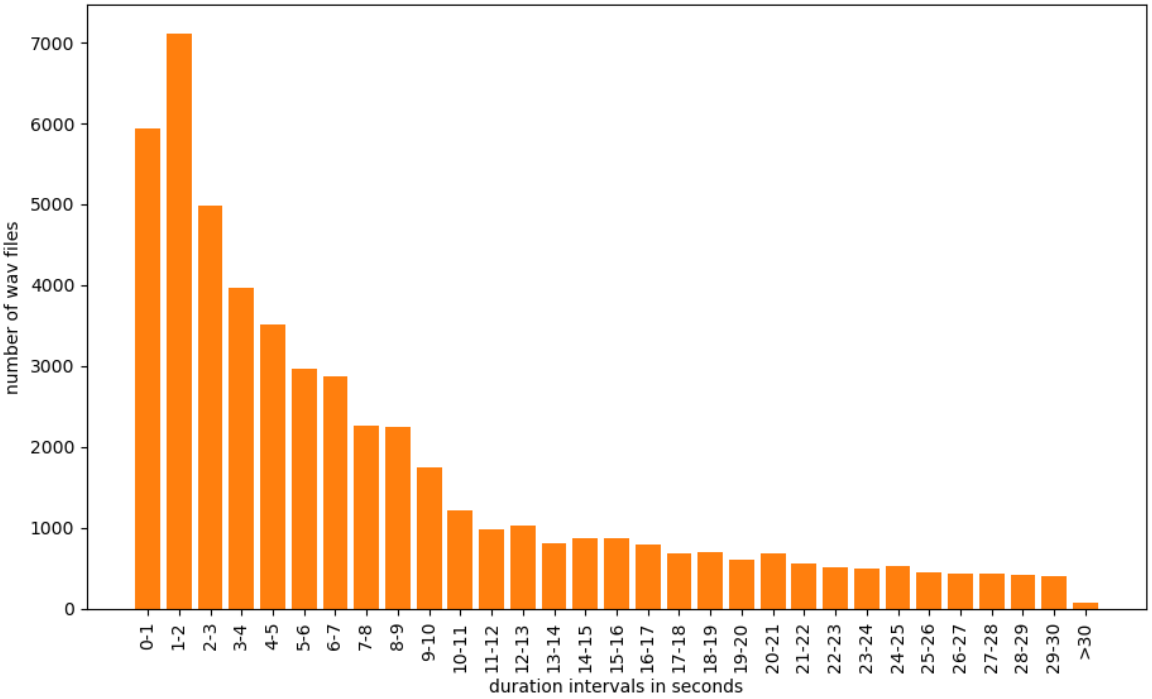


Figure 3: Duration intervals of wav files in FSD50K dataset.

Most of the professional audio recordings can be measured in minutes, while 73.5% of the wav files in FSD50K dataset are less than 10 seconds long. The shortness of the audio files results in some problems for hum detection algorithm.

3. Methodology and Results

FSD50K dataset includes very large variety of short sound samples that are fully tagged with 200 audio classes. Large variety enables us to push the algorithms harder, that makes the tests very robust. The shortness of the sound samples helps speed up the test processes. Tags enables merging the results according to the classes, then most problematic classes can be seen for each audio problem type. These properties make FSD50K a very suitable dataset for testing audio problem detection algorithms.

In this study, all five audio problem detection algorithms and asymmetry calculation algorithm were applied to each audio file in FSD50K dataset. Results were stored for each file, and merged according to 200 audio classifications.

Some audio files are tagged for multiple classes. For example, an audio file for an electrical guitar sample can be tagged as “guitar” and “electrical guitar” separately. In these conditions, results of this audio file were merged for both of the classes.

The most erroneous and least erroneous classes were identified, and the reasons for the results obtained were discussed. Some bugs in the algorithms were detected, fixes and recommendations were proposed.

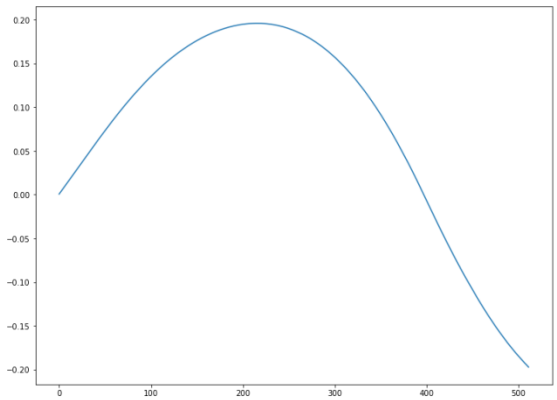
FSD50K dataset does not have ground truths for audio problems in the sound files, which seems as the only disadvantage of this dataset. Because of this, many tests were made manually, and some of the audio files were listened one by one to validate if they include relevant audio problems. In some cases, synthetic problems were generated to test precision of the algorithms.

3.1. Discontinuity Tests

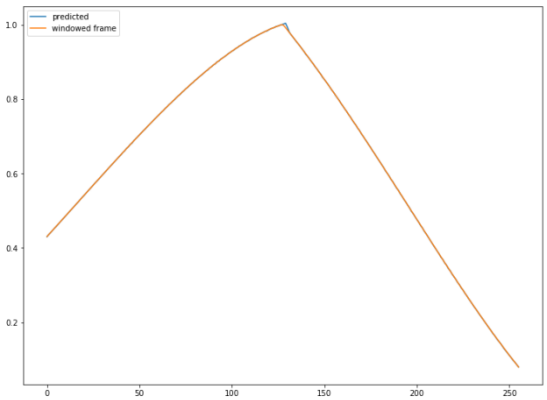
3.1.1. Erroneous Cases

After discontinuity tests were applied, many false positives were detected, mostly in the middle samples of the frames. Algorithm detected discontinuity error even for very smooth parts of the signal.

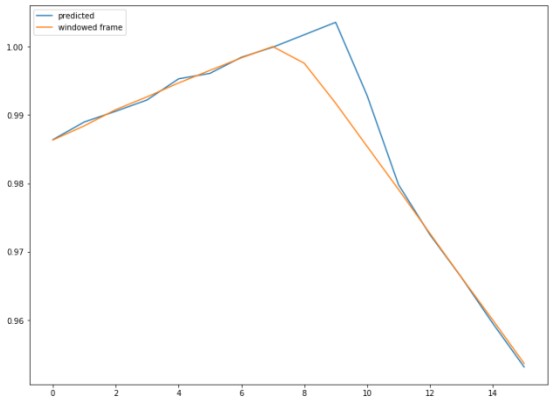
One example for these false positives is given in Figure 4. In Figure 4a, main signal frame takes place. It can be clearly seen that frame is smooth as it can be. The signal frame is windowed, which can be seen with orange in Figure 4b, and predicted windowed signal is shown with blue. In Figure 4c, comparison of the original signal and the predicted signal is given with a closer look. In Figure 4d, subtraction of the original signal from the predicted signal, and the calculated threshold value for this frame is shown.



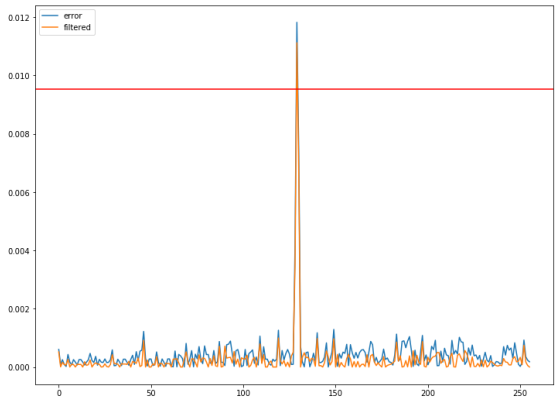
(a) Main smooth signal



(b) Windowed signal, and predicted signal



(c) Windowed and predicted signal (zoom)



(d) Error values and threshold

Figure 4: False positive discontinuity example.

url: https://freesound.org/people/suonho/sounds/58760/

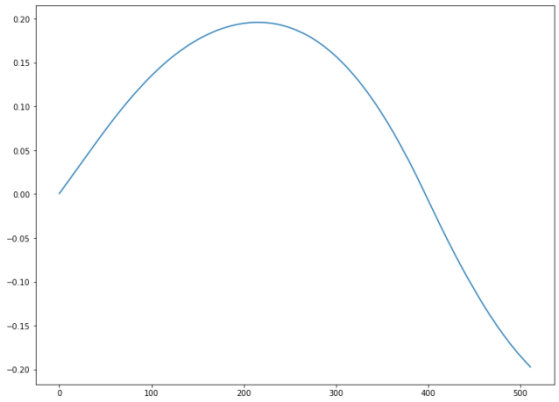
frame: 2, frame size = 512

The difference between windowed and predicted signals can be clearly seen in Figure 4b and Figure 4c. This difference is calculated as error in Figure 4d and the sample points that have error values over threshold is detected as discontinuity.

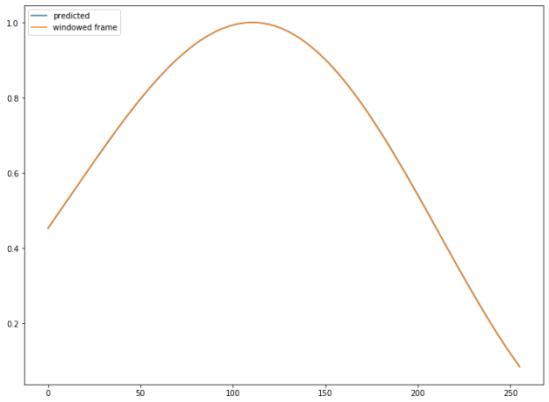
3.1.2. Fixes and Recommendations

After source code of the algorithm was inspected, it was detected that triangular window was used for windowing. By its natural shape, triangular window has a discontinuity in the middle point, and when the original signal is windowed, result frame is usually shaped like a triangle and includes some sort of discontinuity according to its original shape.

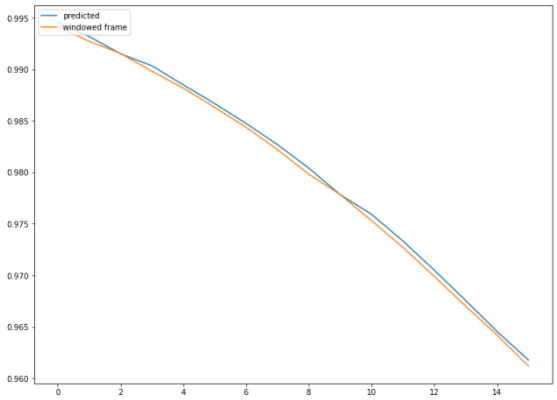
As a fix to this problem, hamming window was used instead of triangular window. In Figure 5, analyze of the same input signal is given, but this time with hamming window.



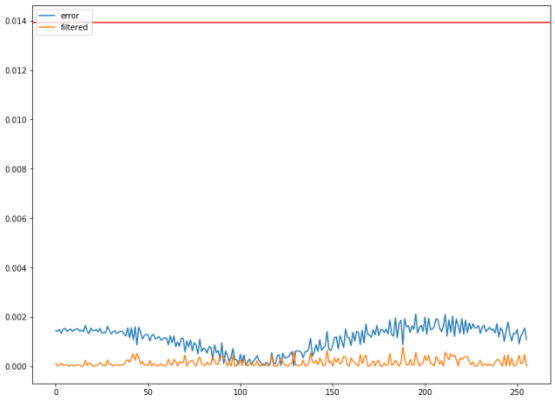
(a) Main smooth signal



(b) Windowed signal, and predicted signal



(c) Windowed and predicted signal (zoom)



(d) Error values and threshold

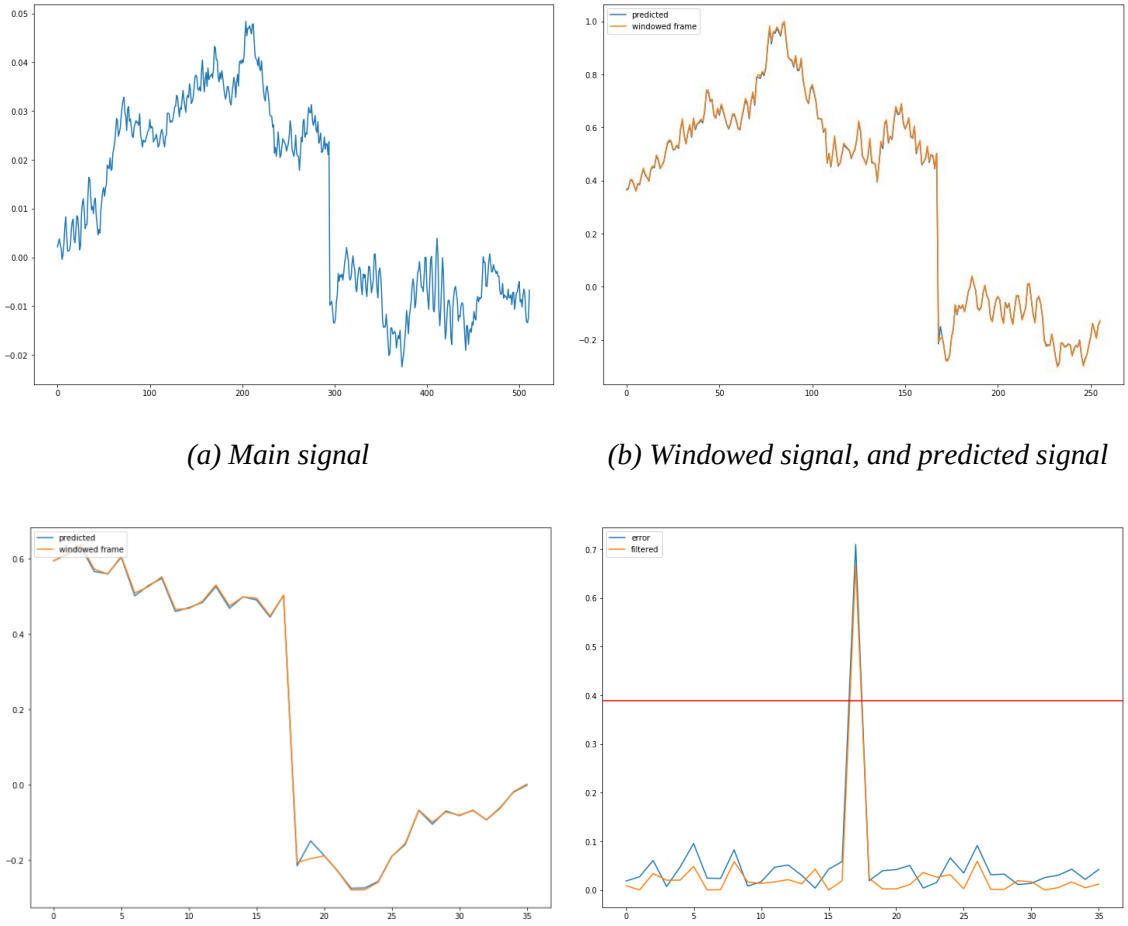
Figure 5: Discontinuity error fixed.

url: https://freesound.org/people/suonho/sounds/58760/

frame: 2, frame size = 512

This time predicted signal could be generated so close to windowed signal and error values did not exceed the threshold value, as expected. After the fix, total discontinuity

count in the whole dataset dropped from 60374 to 15978. In Figure 6, analysis of one frame of real discontinuity is given.



(a) Main signal

(b) Windowed signal, and predicted signal

(c) Windowed and predicted signal (zoom)

(d) Error values and threshold

Figure 6: Discontinuity Example.

url: https://freesound.org/people/Supersciri/sounds/405227/

frame: 2409, frame size: 512

3.1.3. Class results

In Table 1, top 5 classes with highest discontinuity percentage, and in Table 2, 13 classes that have no discontinuity are given. Count column shows the total number of audio files for this class, and discontinuity count column shows the number of audio files that have at least one discontinuity error. Percentage column is calculated as discontinuity_count * 100 / count.

Scratching, cracking, and drum kit are classes that are all expected to have audio samples that can result in discontinuity. By its nature, clipping is also a reason for discontinuity, so some of these discontinuity values may be occurred because of clipping.

Table 1: Top five classes with the highest discontinuity percentages.

Class name	count		discontinuity count	discontinuity percentage
Scratching_(performance_technique)	228		99	43.4
Crackle	220		53	24.1
Drum_kit	341		81	23.8
Boiling	65		14	21.6
Boat_and_Water_vehicle	106		20	18.9

Table 2: Classes with no discontinuity detected.

Class name	count		discontinuity count	discontinuity percentage
Accordion	52		0	0
Chuckle_and_chortle	59		0	0
Crash_cymbal	214		0	0
Cupboard_open_or_close	98		0	0
Finger_snapping	132		0	0
Gasp	58		0	0
Giggle	104		0	0
Glockenspiel	56		0	0
Ratchet_and_pawl	57		0	0
Sawing	118		0	0
Tick	52		0	0
Tick-tock	96		0	0
Typewriter	51		0	0

3.2. Clipping Tests

3.2.1. Erroneous Cases

Clipping algorithm worked as expected for the sounds in FSD50K dataset. No false positive situations were detected. In Figure 7 and Figure 8, audio images of a real clipping example and zoom in version take place. Clipping part can be clearly identified.

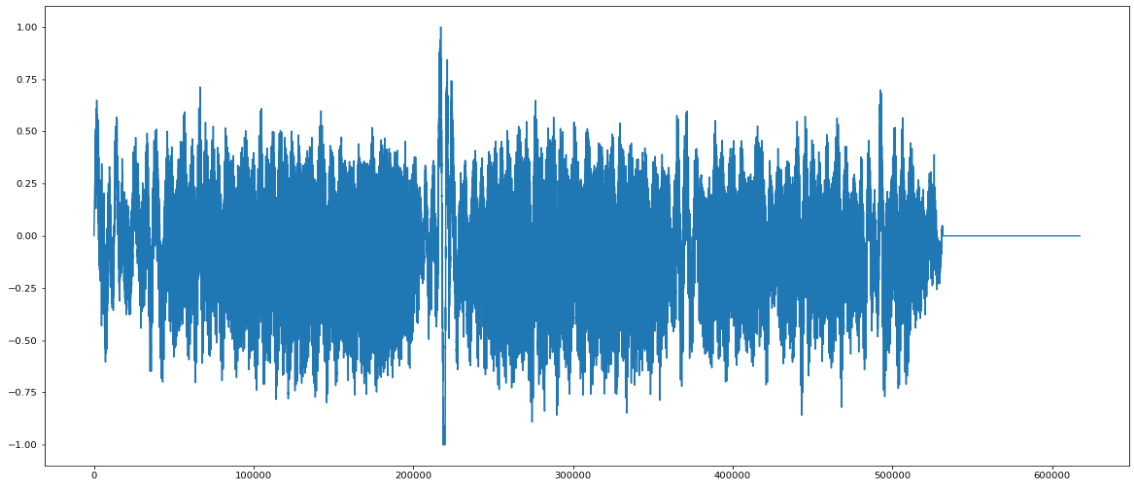


Figure 7: Audio image of a real clipping example.
url: https://freesound.org/people/djgriffin/sounds/21208/

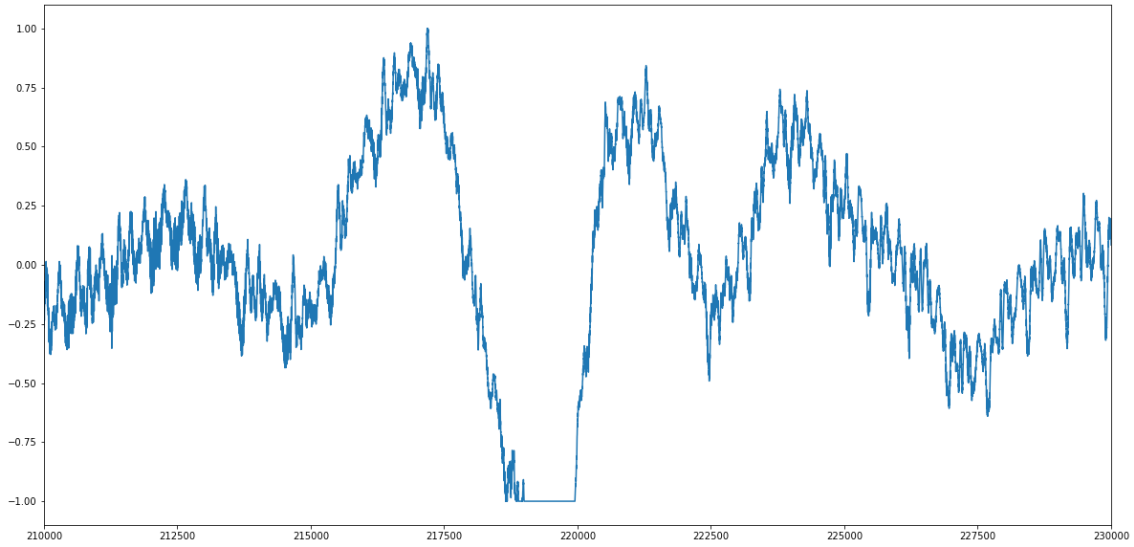


Figure 8: Zoom version of clipping part.
url: https://freesound.org/people/djgriffin/sounds/21208/
samples: 21000 – 23000

3.2.2. Class results

In Table 3, top 5 classes are given according to clipping percentage value. Clipping percentage is the ratio of clipped samples to total samples of all audio files that belong to the given class. In Table 4, top 5 classes that have least clipping percentage are shown.

Table 3: Top five classes with the highest clipping percentages.

Class name	clipping_percentage
Screaming	0.69
Gunshot_and_gunfire	0.60
Siren	0.44
Cheering	0.32
Explosion	0.21

Table 4: Top five classes with the least clipping percentages.

Class name	clipping_percentage
Harmonica	8.5E-06
Crow	1.0E-05
Mechanical_fan	1.4E-05
Chuckle_and_chortle	2.8E-05
Buzz	2.9E-05

"Screaming", "gunshot", "siren", "cheering" and "explosion" classes are all expected to have more clipped samples because of their high amplitude structures.

In Tables 5 – 8, top 5 classes according to audio file percentage that has at least 1 clipping samples, 3 consecutive clipping samples, 44 consecutive clipping samples (which is equal to 1 ms with 44100 sample rate) and 220 consecutive clipping samples (which is equal to 5 ms with 44100 sample rate) can be seen respectively.

Table 5: Top five classes with at least one clipping sample.

Class name	count	1 sample	perc
Gunshot_and_gunfire	348	184	52.9
Fart	533	279	52.3
Shatter	414	167	40.3
Thump_and_thud	299	113	37.8
Sneeze	64	24	37.5

Table 6: Top five classes with at least 3 samples of consecutive clipping samples.

Class name	count	3 samples	perc
Gunshot_and_gunfire	348	139	39.9
Explosion	1122	291	25.9
Sneeze	64	15	23.4
Clapping	378	87	23.0
Shatter	414	92	22.2

Table 7: Top five classes with at least 1ms of consecutive clipping samples.

Class name	count	1ms	perc
Drum_kit	341	22	6.5
Bass_drum	220	10	4.5
Gunshot_and_gunfire	348	15	4.3
Growling	72	3	4.2
Bus	223	9	4.0

Table 8: Top five classes with at least 5ms of consecutive clipping samples.

Class name	count	5ms	Perc
Drum_kit	341	10	2.9
Growling	72	2	2.8
Bus	223	3	1.3
Gunshot_and_gunfire	348	4	1.1
Fixed-wing_aircraft_and_airplane	101	1	0.9

Top 5 classes changed when minimum consequent clipped samples were increased. “Drum kit” was not even in top 5 classes for at least one clipped sample, but it had the highest percentage when minimum clipping duration was increased. In opposite, “fart” class disappeared, which was the second class in Table 5. It can be observed that, if clipping occurs in “drum kit” sounds, it occurs for longer durations, but “fart” sounds have more likely include clipping, but durations are relatively shorter than other classes that includes clipped samples.

3.3. Humming Tests

3.3.1. Erroneous Cases

Hum detection algorithm requires a certain amount of time to detect latter humming noises. This certain time is called “Time Window” and is an input to the function. Because of this limitation, there are minimum durations that hum errors can be detected. For FSD50K dataset, the shortest hum noise that could be detected is 2 sec. In other words, hum noises that are shorter than 2 seconds could not be detected. Also minimum length of the whole audio file, that at least one hum error can be detected was 3.62 sec. These two durations can be accepted as limitations of the algorithm.

As stated in paper [7], false positive ratio for the algorithm gets significantly higher when the duration of the audio file is shorter than 30 sec. In Figure 9, spectrogram of a violin sound at 314 Hz is given. Since the algorithm checks for frequency boundaries and permanency of the sound, this frequency was detected as humming noise. Red line shows the starting and ending times of hum noise. Salience value of this hum noise was given as 1.49, which means a very high saliency.

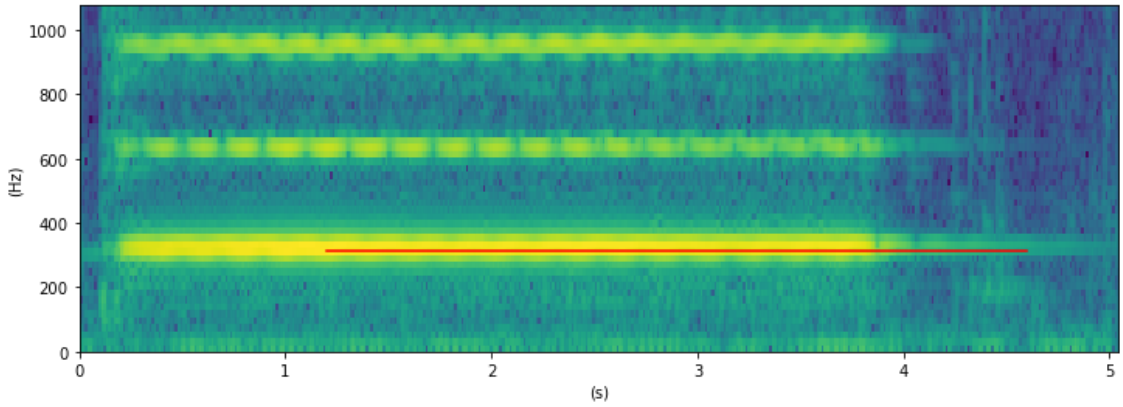


Figure 9: Spectrogram of a violin sound and detected hum range.
url: https://freesound.org/people/clruwe/sounds/119344/

3.3.2. Fixes and Recommendations

There are 77 files in FSD50K dataset that have duration of at least 30 seconds. Humming results of these files were investigated manually, and no false positive cases were detected. According to these results, 30 seconds minimum duration suggestion of the paper [7] was approved, and hum detection algorithm should not be applied to short sounds. But algorithm still finds the hum noises even audio file duration is less than 30 seconds.

In Figure 10, an example spectrogram and detected hum can be seen, where there is hum noise in the audio. Two different hum frequencies were detected at 49.8Hz and 348.05Hz.

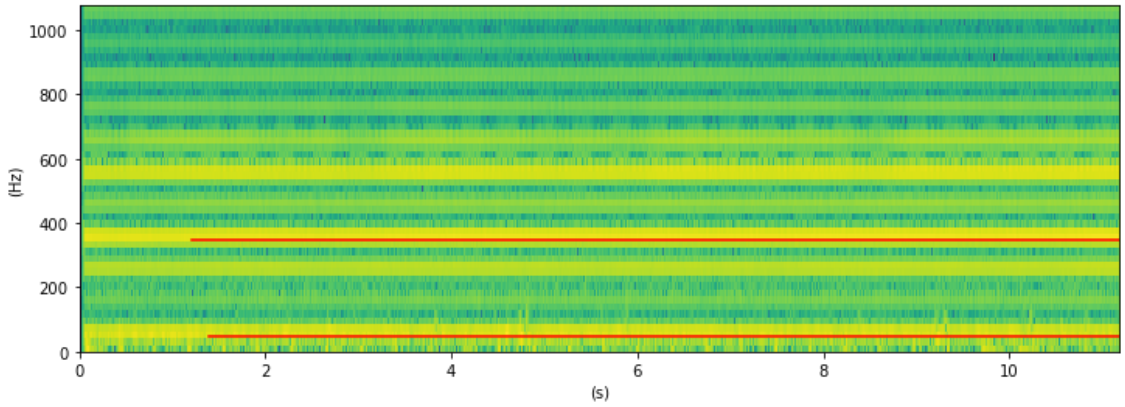


Figure 10: Spectrogram of an environmental sound with hum noises.

3.3.3. Class Results

From Section 2.3, it is known that 44% of the sounds in FSD50K are shorter than 3 seconds, which is shorter than the limitations of the hum detection algorithm. Besides that 99% of the sounds are shorter than 30 seconds, which is under the limitation of the robust working range. According to these values, it can be observed that hum detection algorithm is not reliable and robust for most of the sounds in FSD50K dataset. But, hum detection algorithm still finds results, and not all of them are wrong. Although not completely reliable, inferences can still be made according to results of the classes.

Hum percentage is the ratio of number of audio files that hum is detected to total number of audio files that belong to the given class. In Table 9, top 5 classes are given according to hum percentage. In Table 10, top 5 classes that have least hum percentage are shown.

Table 9: Top five classes having highest hum percentage.

Class name	count	hum_count	hum_percentage
Mechanical_fan	64	55	85.9
Car_passing_by	126	98	77.8
Thunderstorm	469	351	74.8
Thunder	455	338	74.3
Chatter	67	49	73.1

Table 10: Top five classes having lowest hum percentage.

Class name	count	hum_count	hum_percentage
Tabla	52	0	0
Tambourine	229	3	1.3
Fart	533	8	1.5
Cowbell	172	3	1.7
Hi-hat	458	8	1.7

There are many classes which includes humming noises by its nature. “Mechanical fan” and “car passing by” classes are two examples for this issue. In the literature, hum noise is explained as electrical hum in general, which is the result of possible noises that are generated by electrical equipment, cables, microphones, etc., during the recording phase. But the algorithm detects hum errors for the stable frequency components within a certain frequency range, and that lasts long than certain duration. It is questionable if the audio includes hum noise, when the recorded sound itself already includes hum. But, it can be observed that algorithm detects expected classes in the higher rankings.

3.4. Gap Tests

3.4.1. Erroneous Cases

Before starting the gap tests for FSD50K dataset, some simple tests were made with adding gaps to audio files synthetically. It was detected that gap detection algorithm is vulnerable to loudness and amplitude of the audio frame.

A gap detection test was made with an audio file that has amplitude range between 0.25 and -0.25, which uses %25 of the dynamic range. Since the original file do not have a gap error, a synthetic gap was generated synthetically by setting 10000 consecutive samples (0.23 seconds) to zero as shown in Figure 11. Gap detection algorithm was unable to detect this gap.

As mentioned in Chapter 2.2, dynamic ranges of the sounds on FSD50K dataset are nearly randomly distributed. From this perspective, it can be said that, gap detection algorithm will not be able to find the gaps in approximately %25 of these sounds.

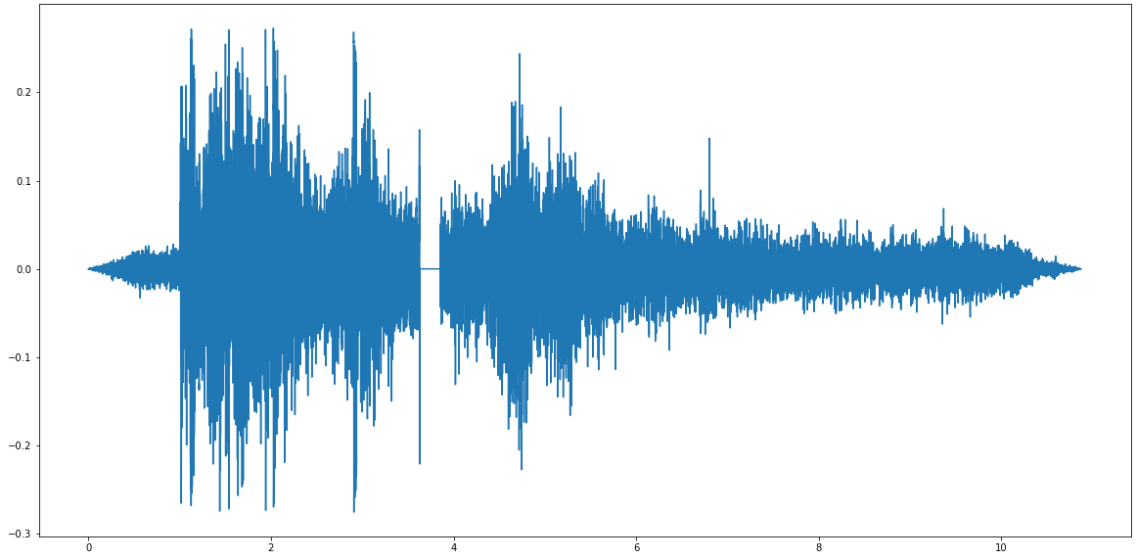


Figure 11: Original audio with gap added.
url: https://freesound.org/search/?q=242981
samples 160000:170000 set to 0

According to research paper [8], there are two more cases that gap detection algorithm fails to operate. These cases are audio recordings of electronic music and human speech. Amplitude envelope of electronic music instruments do not have to decay slowly. They can have instant stops, so gap detection algorithm can detect normal silences as gaps. Human voice also has a similar characteristic.

3.4.2. Fixes and Recommendations

When the audio file in Figure 11 was normalized, Figure 12 was generated. The range of the sample values was between -1.0 and 1.0 in Figure 12. After normalizing the audio file, gap detection algorithm could find the gap. Vertical red line shows the start, and vertical green line shows the end point of the detected gap. As a fix for the test, all audio files were normalized before gap detection algorithm was applied.

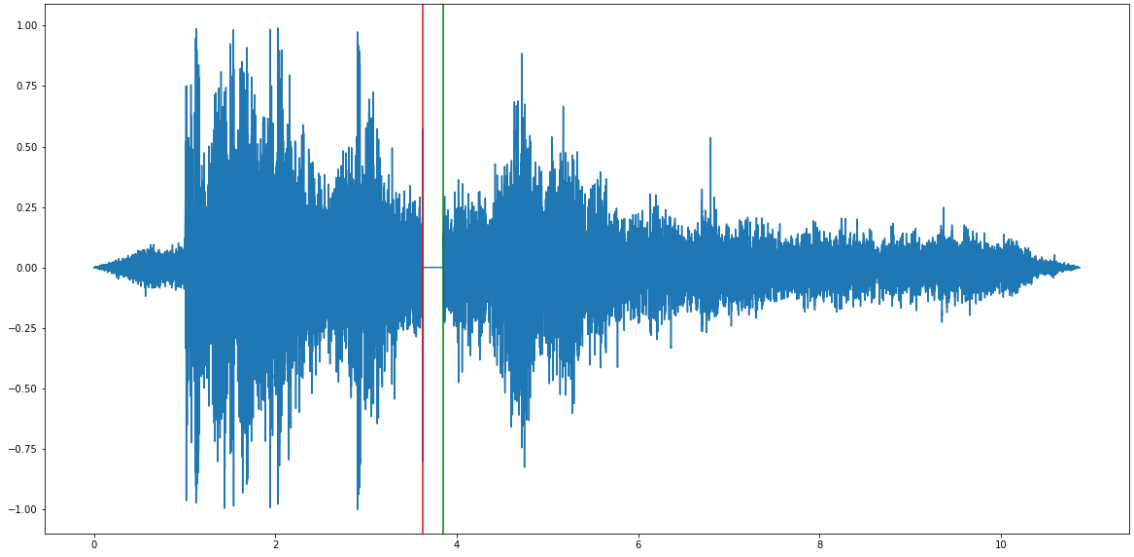


Figure 12: Normalized audio with gap added.

url: https://freesound.org/search/?q=242981

3.4.3. Class results

Table 11 shows the top five classes having the highest gap sample percentage. Gap sample percentage is the ratio of the total number of samples that are detected as gap, and the total number of samples in the audio file. Table 12 shows the top five classes having the biggest gap sizes in seconds.

Table 11: Top five classes having highest gap sample percentage.

Class name	count	gap_sample_percentage
Scratching_(performance_technique)	228	8.2
Telephone	520	3.5
Alarm	1280	2.7
Scissors	106	1.6
Stream	214	1.5

Table 12: Top five classes having biggest gap sizes.

Class name	count	biggest_gap_size
Scissors	106	2.9
Domestic_sounds_and_home_sounds	4711	2.9
Telephone	520	2.9
Alarm	1280	2.9
Harp	177	2.0

Gap percentage is the ratio of number of audio files that at least one gap is detected, to total number of audio files that belong to the given class. In Table 13, top 5 classes are given according to gap percentage.

Table 13: Top five classes having highest gap percentage.

Class name	count	gap count	gap percentage
Scratching_(performance_technique)	228	89	39.0
Telephone	520	84	16.2
Ratchet_and_pawl	57	9	15.8
Speech_synthesizer	59	9	15.3
Alarm	1280	150	11.7

All classes in Tables 11-13 belong to objects or events that generate intermittent sounds, so there is no unexpected entry in this list. A detailed analysis for the class “telephone”, which takes second place in “gap percentage list”, and takes third place in “biggest gap size list”, is shown below.

In Figure 13, an audio image for a “free call on telephone” is given. Two gaps were detected in the first half of the file. The following spaces were not detected as gaps, since the audio decays slowly at those parts, so these parts were detected as normal silences.

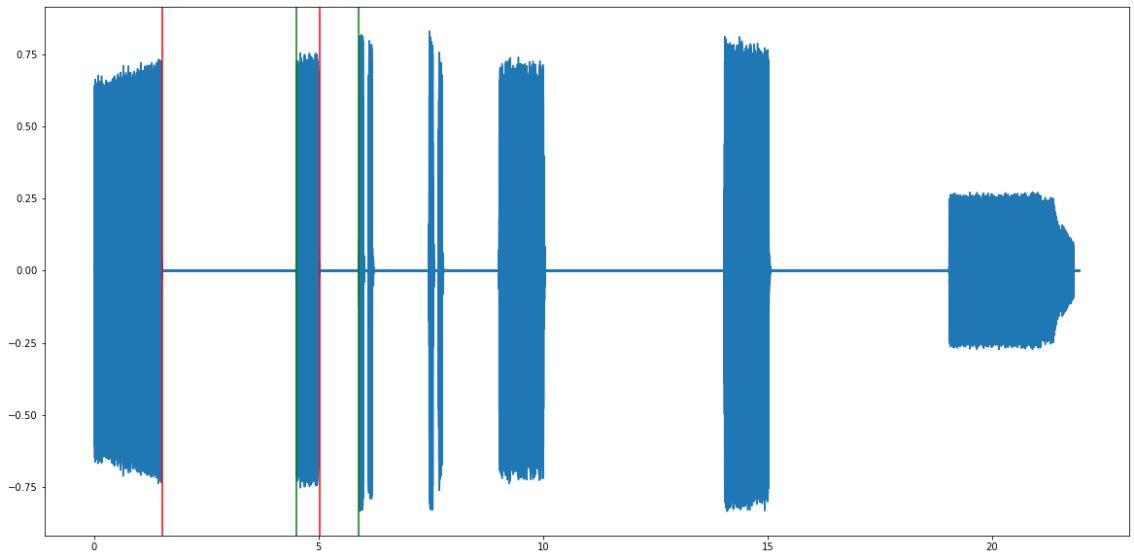


Figure 13: Audio image of a free call to a random free number containing only beeps. url: https://freesound.org/people/Felfa/sounds/188695/

In Figure 14, some of the empty parts in the audio file that were detected as normal silences are shown. Because of the decay parts at the end of the audio parts, empty parts were not detected as gap problems, but a normal silence part in the audio.

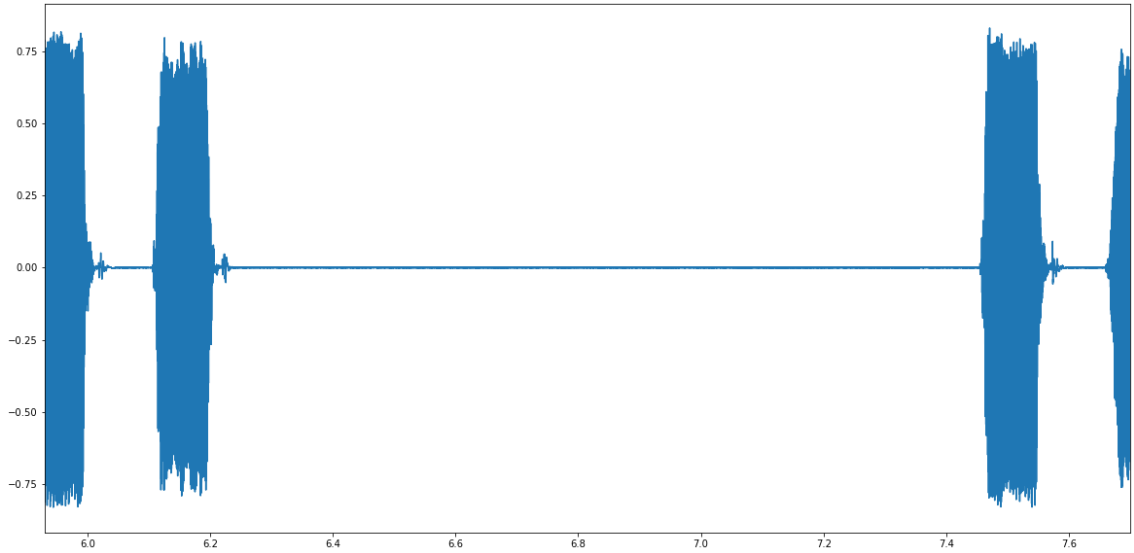


Figure 14: Silence parts in the audio file.

In Figure 15, detected gap between seconds 5.0 and 5.9 is shown. Decay part is much shorter and sudden, so this empty part was detected as a gap.

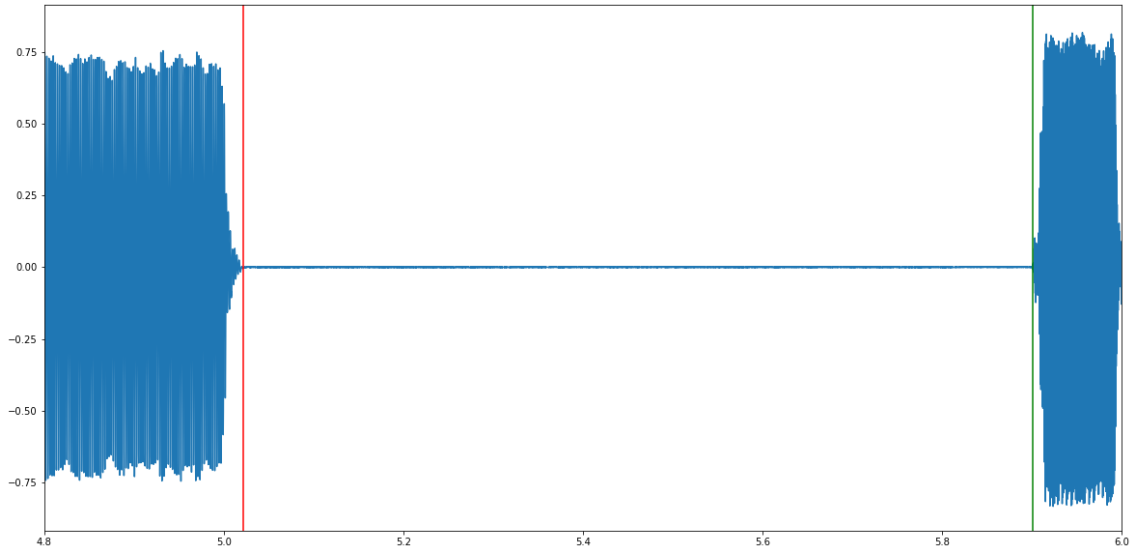


Figure 15: Detected gap between seconds 5.0 – 5.9

3.5. Click and Pop Tests

3.5.1. Erroneous Cases

Click and Pop algorithm worked as expected for the sounds in FSD50K dataset. No false positive situations were detected.

In the Figure 16, an audio image of a real click and pop example of a “rain drop” class takes place. Click part can be clearly identified in Figure 17.

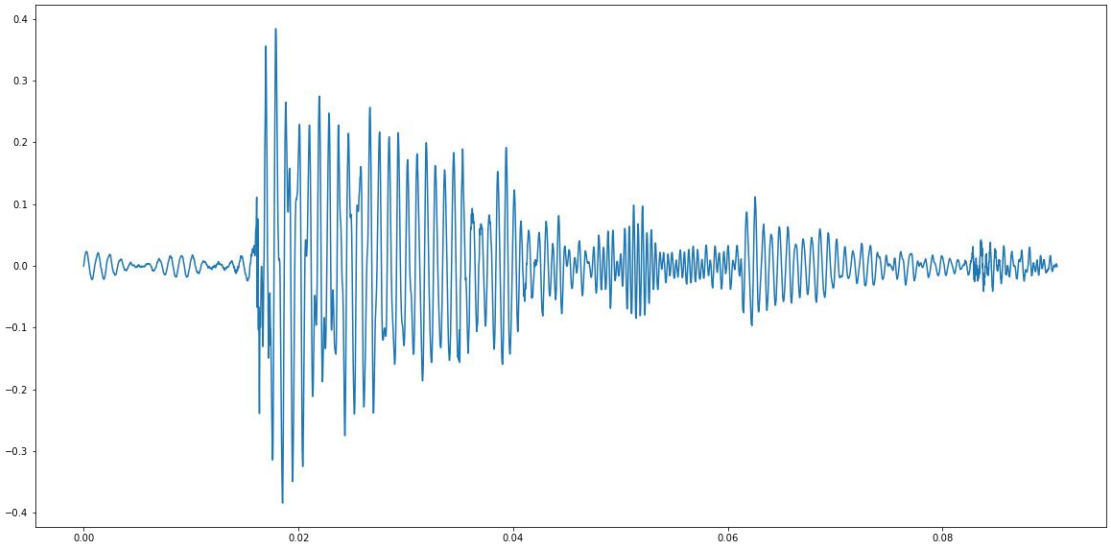


Figure 16: Audio image of a click and pop example.

url: https://freesound.org/people/melarancida/sounds/47385/

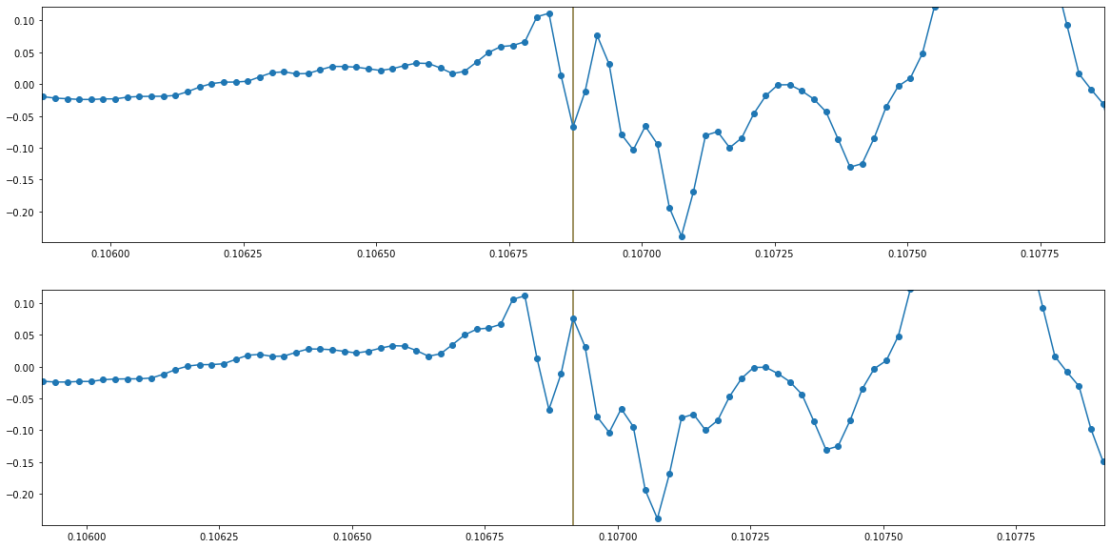


Figure 17: Consecutive click detection.

url: https://freesound.org/people/melarancida/sounds/47385/

3.5.2. Class results

In Table 14, top five classes are given according to click percentage values. Click percentage is the ratio of number of audio files that at least one click is detected, to total number of audio files that belong to the given class. In Table 15, top five classes that have the least click percentages are given.

Table 14: Top five classes with highest click percentages.

Class name	count	click_count	click_perc
Crackle	220	187	85
Crack	118	98	83.1
Ratchet_and_pawl	57	47	82.4
Drum_kit	341	265	77.7
Packing_tape_and_duct_tape	70	49	70.0

Table 15: Top five classes with least click percentages.

Class name	count	click_count	click_perc
Cowbell	172	3	1.7
Glockenspiel	56	1	1.8
Sigh	75	2	2.7
Harmonica	165	6	3.6
Chicken_and_rooster	138	7	5.1

3.6. Asymmetry Tests

3.6.1. Asymmetry Examples

Asymmetry is not an audio problem; it is only a parameter according to audio envelope of the recorded sound. However in some cases, like gain controls, it can lead to clipping or distortion problems.

In practical, in other words in common recordings, two mostly seen asymmetry examples are brass instruments and female speech. Audio images for wav files for both classes are given in Figure 18 and Figure 19.

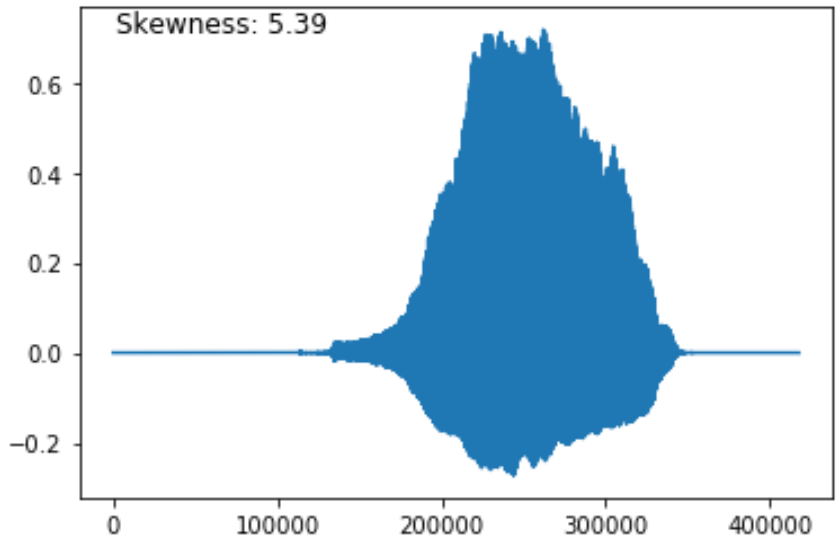


Figure 18: Audio image of brass instrument.
url: https://freesound.org/people/MTG/sounds/357576/

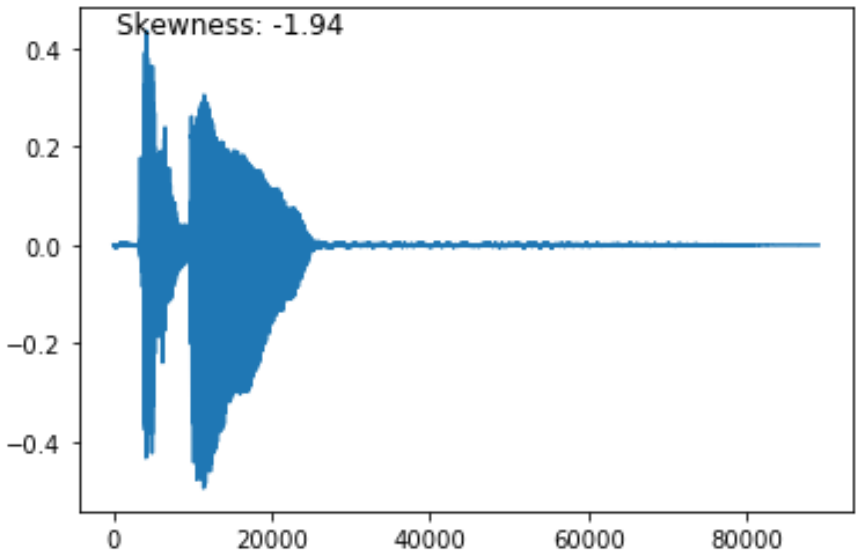


Figure 19: Audio image of female speech.
url: https://freesound.org/people/tim.kahn/sounds/67750/

In addition to these sounds, audio recordings including intermittent attacks can also have high skewness values, since attacks generate samples mostly on the same side of the axis. In Table 16, top five classes with the highest average skewness values are given. All the top classes are sounds having intermittent attacks.

Figure 20 shows the effect of attacks to asymmetry, which is a sound of a “Tapping on a glass bottle with a drumstick”.

Table 16: Top five classes with highest average skewness values.

Class name	Average_skewness
Tick-tock	48.0
Crackle	28.7
Crack	22.3
Fireworks	21.7
Clock	21.1

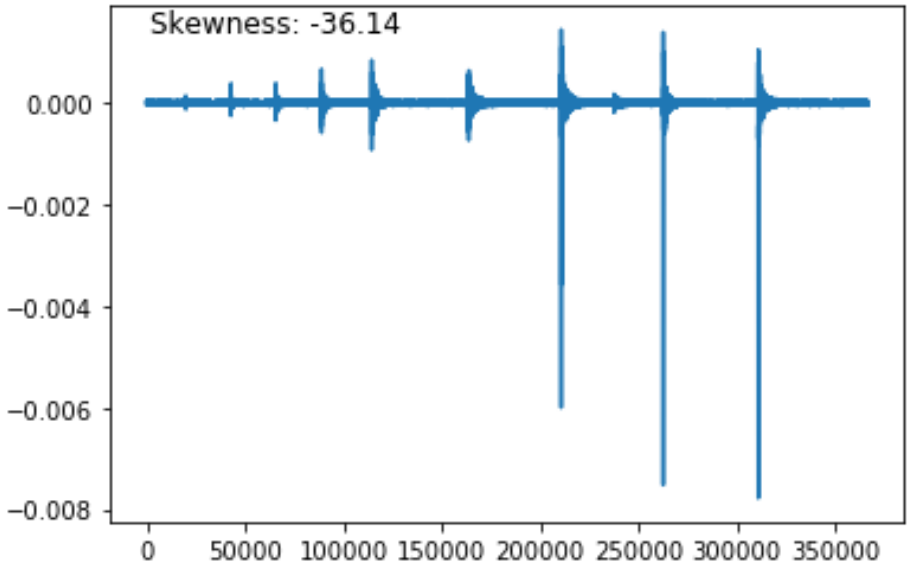


Figure 20: Tapping on a glass bottle with a drumstick.
url: https://freesound.org/people/rstthedave/sounds/186531/

4. Audio Analyzer in Freesound

Freesound is a web platform that can list audio results according to different filters. Users can filter sounds according to license type, file type, sample rate, bit depth, bit rate, or channel counts. In addition to these filters, a new filter according to whether a file has audio problem or not, according to five audio problem types in this study, was added.

After five audio problem algorithms were tested separately, an audio analyzer was developed that analyzes sound files with all five types of audio problem detection algorithms [18]. The analyzer was applied to all sound files in Freesound Platform, and the filter can be used as a search parameter.

A file is accepted as problematic if the file:

- includes at least one discontinuity error,
- includes at least one gap error,
- includes at least three consecutive click samples,
- is at least 30 seconds long, and includes a hum problem with a salience value more than 1.0,
- includes at least one clipping error.

According to the rules above, heat map of FSD50K dataset can be seen in Table 17 in Appendix 1. In Table 17, “name” column shows the 200 classes for FSD50K, “file count” column shows the number of all audio files that falls into that class. “disc perc”, “clip perc”, “gap perc” and “click perc” columns shows the percentage of files in the class that have at least one error according to the rules mentioned above. “error perc” column shows the percentage of the files that have at least one kind of error, so that is tagged as erroneous. In the table, humming error does not take place, since there are 77 files that are longer than 30 seconds, and none of these files have humming error having salience more than 1.0. FSD50K dataset includes nearly %8 of the files that are uploaded to Freesound Platform up to now, so it can be an adequate map to guess the overall Freesound heat map.

5. Conclusions

In this study, audio problem detection algorithms in Essentia library were applied to FSD50K dataset, which is constructed with 50K sounds from Freesound web platform.

First of all, FSD50K dataset was analyzed. Properties like duration histograms and dynamic range histograms were calculated, sound taxonomies were introduced. Properties of the audio files in the dataset are different from usual music files, so it was a robustness test for the audio problem detection algorithms.

Five common audio problem types were selected, which were discontinuity, clipping, hum, gap, and click. Detection algorithms of these problem types were applied to dataset separately. A bug in discontinuity detection algorithm was identified and fixed. Some limitations were recommended for hum detection algorithm. A solution was proposed for better performance in gap detection algorithm.

For each audio problem type, problem percentages were calculated for each sound class in the dataset. As a result, the most problematic sound classes were identified, results were discussed separately.

Freesound is visited and used by many people including content or music creators. Artists densely use the platform to find sounds and use them in their own studies, thanks to the free usage and large variety of the sounds. But these artists usually want the sound samples to be problem free, so they can easily use the sound without further post processing. After the tests, an audio analyzer was added to Freesound that enables users to filter sounds according to whether if the file includes audio problem or not, according to five audio problem types selected for this study. The analyzer uses thresholds and limitations that were tested, found or recommended in this study.

In the study, five of the audio problems were analyzed and implemented as a problem detector for Freesound. As a future work, other audio problem types can be added, and detection algorithms can be tested, results can be analyzed, so audio analyzer can detect other types of problems as well.

Another future work can be investigating the results of audio analyzer in Freesound and analyzing the thresholds used in the algorithms. For example, a sound file is accepted to

be problematic if it includes at least three consecutive clipping samples, or includes a hum with salience larger than 1.0. But after investigating the audio analyzer results and user feedbacks, these thresholds can be fine-tuned, which can lead to a better user experience.

6. List of Figures

Figure 1: Histogram comparison for max(abs(audio)) values calculated for each audio file. .. 10
Figure 2: Histogram comparison dynamic range values calculated for each audio file. 10
Figure 3: Duration intervals of wav files in FSD50K dataset. 11
Figure 4: False positive discontinuity example. 13
Figure 5: Discontinuity error fixed. 14
Figure 6: Discontinuity Example. 15
Figure 7: Audio image of a real clipping example. 17
Figure 8: Zoom version of clipping part. 17
Figure 9: Spectrogram of a violin sound and detected hum range. 20
Figure 10: Spectrogram of an environmental sound with hum noises. 21
Figure 11: Original audio with gap added. 23
Figure 12: Normalized audio with gap added. 24
Figure 13: Audio image of a free call to a random free number containing only beeps. 25
Figure 14: Silence parts in the audio file. 26
Figure 15: Detected gap between seconds 5.0 – 5.9 26
Figure 16: Audio image of a click and pop example. 27
Figure 17: Consecutive click detection. 27
Figure 18: Audio image of brass instrument. 29
Figure 19: Audio image of female speech. 29
Figure 20: Tapping on a glass bottle with a drumstick. 30

7. List of Tables

Table 1: Top five classes with the highest discontinuity percentages. 16
Table 2: Classes with no discontinuity detected. 16
Table 3: Top five classes with the highest clipping percentages. 18
Table 4: Top five classes with the least clipping percentages. 18
Table 5: Top five classes with at least one clipping sample. 18
Table 6: Top five classes with at least 3 samples of consecutive clipping samples. 18
Table 7: Top five classes with at least 1ms of consecutive clipping samples. 19
Table 8: Top five classes with at least 5ms of consecutive clipping samples. 19
Table 9: Top five classes having highest hum percentage. 21
Table 10: Top five classes having lowest hum percentage. 22
Table 11: Top five classes having highest gap sample percentage. 24
Table 12: Top five classes having biggest gap sizes. 24
Table 13: Top five classes having highest gap percentage. 25
Table 14: Top five classes with highest click percentages. 28
Table 15: Top five classes with least click percentages. 28
Table 16: Top five classes with highest average skewness values. 30
Table 17: Heat Map for FSD50K error percentages. 38

8. Bibliography

1. Freesound. https://freesound.org/ (2023).
2. Bogdanov, D. et al. Essentia: An audio analysis library for music information retrieval. in *the 14th Conference of the International Society for Music Information Retrieval Conference (ISMIR'13), Curitiba, Brazil, November 4-8, 2013* (eds. Britto A., Gouyon F., & Dixon S.) 493–498, (2013).
3. Alonso-Jiménez, P., Joglar-Ongay, L., Serra, X., & Bogdanov, D. Automatic detection of audio problems for quality control in digital music distribution. in *146th Convention of the Audio Engineering Society, Dublin, Ireland, March 20-23, 2019* (2019).
4. Algortihms reference. https://essentia.upf.edu/algorithms_reference.html (2023).
5. Rabiner, L. R. & Schafer, R. W. *Digital Processing of Speech Signals*. (Prentice-Hall, 1978).
6. International Telecommunication Union. *Recommendation ITU-R BS.1770-4: Algorithms to Measure Audio Programme Loudness and True-Peak Audio Level*. (2017).
7. Brandt, M. & Bitzer, J. Automatic detection of hum in audio signals. *Journal of the Audio Engineering Society* 62, 584–595 (2014).
8. Mühlbauer, R. Automatic audio defect detection. (Johannes Kepler Universität, Linz, 2010).
9. Impulsive Noise: Modelling, Detection and Removal. in *Advanced Digital Signal Processing and Noise Reduction, Fourth Edition* (ed. Vaseghi, S. V.) 341-358 (John Wiley & Sons, 2008).
10. Wolff, D., Mignot, R. & Roebel, A. Audio defect detection in music with deep networks. Preprint at https://arxiv.org/abs/2202.05718 (2022).
11. Zwillinger, D. & Kokoska, S. (2000). *CRC Standard Probability and Statistics Tables and Formulae.* (Chapman & Hall, 2000).

12. Ignasi A. A. Automatic detection of audio defects in personal music collections. (Universitat Pompeu Fabra, 2016).

13. Crespo, V. B. Audio problems detection on sound collections. (Universitat Pompeu Fabra, 2019).

14. Freesound audio tagging 2019, automatically recognize sounds and apply tags of varying natures. https://www.kaggle.com/c/ freesound-audio-tagging-2019.

15. Fonseca, E., Favory, X., Pons, J., Font, F., & Serra X. FSD50K: An open dataset of human-labeled sound events. Preprint at https://arxiv.org/abs/2010.00475 (2021).

16. Gemmeke, J. F. et al. Audio Set: An ontology and human-labeled dataset for audio events. in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA,* 776-780 (2017).

17. Bogdanov, D., Lizarraga-Seijas, X., Alonso-Jiménez, P., & Serra X. MusAV: A dataset of relative arousal-valence annotations for validation of audio models. in *Proceedings of the 23rd International Society for Music Information Retrieval Conference (ISMIR 2022), Bengaluru, India, December 4-8, 2022,* (eds. Rao, P., Murthy, H., Srinivasamurthy, A., Bittner, R., Caro Repetto, R., Goto, M., Serra, X., & Miron M.) 650 - 658 (2022).

18. Audio analyzer. https://github.com/MTG/freesound-audio-analyzers/tree/main/fs-essentia-problem-detection_v1

9. Appendices

9.1. Appendix 1

Table 17: Heat Map for FSD50K error percentages.

name	file count	disc perc	clip perc	gap perc	click perc	error perc
Accelerating_and_revving_and_vroom	161	5.0	6.8	0.6	11.8	20.5
Accordion	52	0.0	1.9	0.0	9.6	11.5
Acoustic_guitar	538	3.0	7.1	0.0	29.7	31.4
Aircraft	184	10.3	9.8	0.5	17.4	28.3
Alarm	1280	12.0	5.8	11.7	36.8	42.3
Animal	3275	4.6	4.0	1.0	13.7	18.0
Applause	400	7.0	11.0	0.0	48.3	51.5
Bark	414	2.2	7.2	0.2	5.8	11.8
Bass_drum	220	9.5	16.8	1.8	26.4	36.4
Bass_guitar	322	10.2	0.3	0.0	11.2	17.7
Bathtub_(filling_or_washing)	149	1.3	0.0	0.0	20.1	20.1
Bell	798	2.4	1.9	0.1	20.4	21.8
Bicycle	211	3.3	1.9	0.5	37.4	38.4
Bicycle_bell	73	1.4	1.4	0.0	52.1	53.4
Bird	1189	4.5	2.2	0.8	13.3	16.8
Bird_vocalization_and_bird_call_and_bird_song	502	3.8	2.6	1.4	11.8	16.3
Boat_and_Water_vehicle	106	18.9	8.5	1.9	27.4	34.0
Boiling	65	21.5	1.5	0.0	30.8	35.4
Boom	167	16.8	19.8	0.6	31.7	42.5
Bowed_string_instrument	1924	0.6	0.2	0.0	7.3	8.0
Brass_instrument	949	1.3	0.3	0.0	9.0	9.6
Breathing	431	1.4	1.2	0.2	7.7	8.8
Burping_and_eructation	219	0.9	5.5	0.5	28.3	31.1
Bus	223	12.1	8.5	0.0	23.3	29.6
Buzz	80	3.8	0.0	0.0	16.3	18.8
Camera	300	1.0	4.3	2.7	45.7	48.0
Car	673	5.9	3.6	0.3	16.8	20.8
Car_passing_by	126	2.4	1.6	0.0	8.7	11.1
Cat	303	6.6	4.3	1.0	18.8	22.4
Chatter	67	1.5	3.0	0.0	13.4	14.9
Cheering	217	12.0	18.9	0.0	33.6	43.3
Chewing_and_mastication	181	3.9	6.6	1.1	56.4	59.1

Chicken_and_rooster	138	2.9	3.6	0.0	5.1	7.2
Child_speech_and_kid_speaking	158	1.9	0.6	0.0	11.4	12.7
Chime	166	3.0	2.4	0.0	22.9	24.1
Chink_and_clink	265	1.1	6.8	0.8	51.3	53.6
Chirp_and_tweet	194	1.0	1.0	2.1	8.2	11.3
Chuckle_and_chortle	59	0.0	0.0	1.7	10.2	11.9
Church_bell	81	4.9	0.0	0.0	13.6	14.8
Clapping	378	1.6	23.0	1.3	35.7	49.2
Clock	252	0.8	2.0	2.0	30.6	32.1
Coin_(dropping)	437	0.9	8.2	0.7	47.4	50.8
Computer_keyboard	130	3.1	6.9	1.5	58.5	60.8
Conversation	65	1.5	3.1	0.0	20.0	24.6
Cough	279	2.2	13.3	0.0	20.4	30.5
Cowbell	172	0.6	0.0	0.6	1.7	1.7
Crack	118	0.8	17.8	1.7	83.1	85.6
Crackle	220	24.1	3.2	0.9	85.0	85.0
Crash_cymbal	214	0.0	0.5	0.0	6.1	6.5
Cricket	176	1.7	2.3	4.5	10.8	16.5
Crow	75	8.0	0.0	0.0	18.7	21.3
Crowd	282	6.4	4.3	0.0	26.6	29.8
Crumpling_and_crinkling	201	3.0	4.5	1.5	69.7	71.1
Crushing	190	0.5	8.4	3.2	57.9	62.6
Crying_and_sobbing	109	2.8	9.2	2.8	15.6	24.8
Cupboard_open_or_close	98	0.0	5.1	0.0	25.5	29.6
Cutlery_and_silverware	274	0.4	1.8	0.0	32.8	34.7
Cymbal	669	0.4	0.7	0.0	7.3	7.6
Dishes_and_pots_and_pans	330	1.2	7.9	0.0	33.9	41.2
Dog	752	3.6	6.3	0.3	11.0	16.9
Domestic_animals_and_pets	1054	4.5	5.7	0.5	13.3	18.5
Domestic_sounds_and_home_sounds	4711	2.7	5.6	0.7	35.6	38.7
Door	1056	2.0	7.5	0.5	28.2	31.3
Doorbell	107	0.9	2.8	0.0	20.6	21.5
Drawer_open_or_close	151	1.3	6.6	0.0	27.8	31.1
Drill	169	1.2	3.6	1.2	20.1	20.7
Drip	232	4.3	2.6	1.3	51.3	53.0
Drum	1204	4.3	5.7	0.6	19.4	22.4
Drum_kit	341	23.8	17.9	3.8	77.7	81.5
Electric_guitar	506	7.9	1.4	0.4	17.2	21.1
Engine	854	6.6	5.4	0.4	19.2	23.8
Engine_starting	144	8.3	8.3	0.7	24.3	30.6
Explosion	1122	12.8	25.9	1.2	47.9	58.6
Fart	533	3.2	11.3	0.4	35.6	41.7
Female_singing	137	2.2	0.0	0.7	9.5	10.2

Female_speech_and_woman_speaking	393	4.1	1.5	1.0	42.7	43.8
Fill_(with_liquid)	90	5.6	5.6	1.1	41.1	43.3
Finger_snapping	132	0.0	2.3	0.0	57.6	58.3
Fire	385	7.5	9.9	0.5	52.5	57.7
Fireworks	402	14.4	15.9	0.2	63.9	66.2
Fixed-wing_aircraft_and_airplane	101	11.9	11.9	0.0	13.9	27.7
Fowl	224	3.6	4.0	0.0	8.9	10.7
Frog	72	2.8	1.4	2.8	19.4	22.2
Frying_(food)	66	4.5	4.5	0.0	30.3	33.3
Gasp	58	0.0	5.2	0.0	6.9	12.1
Giggle	104	0.0	5.8	1.9	12.5	15.4
Glass	974	1.1	12.3	0.2	41.0	47.1
Glockenspiel	56	0.0	0.0	0.0	1.8	1.8
Gong	206	2.4	1.9	1.0	10.2	13.6
Growling	72	11.1	11.1	1.4	18.1	25.0
Guitar	1821	8.1	3.4	0.4	22.1	25.8
Gull_and_seagull	72	9.7	2.8	0.0	20.8	25.0
Gunshot_and_gunfire	348	9.8	39.9	2.3	43.4	62.4
Gurgling	133	6.8	3.0	0.0	30.1	33.8
Hammer	165	3.0	10.9	0.0	52.1	57.0
Hands	573	1.0	16.9	0.9	42.4	51.5
Harmonica	165	0.6	0.0	0.0	3.6	4.2
Harp	177	1.7	0.6	0.6	9.6	10.2
Hi-hat	458	0.7	0.9	0.0	7.9	8.1
Hiss	164	1.8	4.9	1.2	18.3	20.7
Human_group_actions	792	6.8	8.8	0.0	35.0	38.9
Human_voice	4020	4.0	6.3	2.4	22.8	27.6
Idling	107	10.3	1.9	0.9	15.9	18.7
Insect	383	2.9	2.1	2.9	13.1	17.2
Keyboard_(musical)	1428	11.1	2.2	0.8	22.8	26.0
Keys_jangling	169	0.6	3.6	0.0	40.2	42.0
Knock	270	2.2	11.1	0.7	42.6	44.4
Laughter	933	2.5	8.5	1.1	12.9	19.0
Liquid	1125	5.1	2.9	0.4	40.7	42.4
Livestock_and_farm_animals_and_working_animals	515	5.4	5.0	0.4	13.8	17.7
Male_singing	80	1.3	1.3	0.0	10.0	11.3
Male_speech_and_man_speaking	391	2.6	4.3	1.0	30.2	33.5
Mallet_percussion	287	1.7	1.4	0.0	10.8	12.2
Marimba_and_xylophone	171	2.9	2.3	0.0	16.4	18.7
Mechanical_fan	64	18.8	0.0	0.0	23.4	26.6
Mechanisms	1073	3.1	3.4	2.2	36.3	38.5
Meow	172	4.7	3.5	0.6	12.8	16.3
Microwave_oven	140	4.3	8.6	0.0	28.6	35.0

Motor_vehicle_(road)	1399	7.2	4.6	0.4	20.2	24.4
Motorcycle	170	5.3	4.1	0.0	12.9	17.6
Music	12767	4.9	2.3	1.0	15.0	17.1
Musical_instrument	12763	4.9	2.3	1.0	15.0	17.1
Ocean	239	5.0	3.8	0.0	13.8	18.0
Organ	313	10.9	1.0	1.0	23.0	24.9
Packing_tape_and_duct_tape	70	2.9	4.3	5.7	70.0	71.4
Percussion	3152	4.3	4.3	0.6	18.9	21.1
Piano	779	9.4	1.3	0.3	17.8	20.7
Plucked_string_instrument	1916	7.8	3.3	0.4	21.7	25.3
Pour	154	5.8	1.3	0.0	52.6	53.2
Power_tool	171	1.2	3.5	1.2	19.9	20.5
Printer	100	5.0	0.0	0.0	22.0	23.0
Purr	68	7.4	4.4	0.0	32.4	35.3
Race_car_and_auto_racing	72	15.3	1.4	0.0	15.3	22.2
Rail_transport	637	10.5	14.4	0.2	18.7	29.5
Rain	500	6.4	4.4	0.6	29.0	34.6
Raindrop	119	1.7	1.7	2.5	52.9	53.8
Ratchet_and_pawl	57	0.0	7.0	15.8	82.5	84.2
Rattle	173	3.5	4.0	0.6	45.7	47.4
Rattle_(instrument)	253	1.6	3.2	0.4	21.3	23.3
Respiratory_sounds	822	1.7	7.2	0.1	12.3	18.2
Ringtone	97	17.5	4.1	6.2	43.3	51.5
Run	241	12.9	6.6	0.0	60.2	61.0
Sawing	118	0.0	2.5	0.0	13.6	15.3
Scissors	106	1.9	11.3	4.7	37.7	45.3
Scratching_(performance_technique)	228	43.4	8.8	39.0	67.5	82.9
Screaming	254	3.1	15.0	0.4	15.4	27.6
Screech	86	7.0	4.7	0.0	27.9	31.4
Shatter	414	1.2	22.2	0.0	40.6	52.7
Shout	216	5.6	5.6	0.0	21.8	25.9
Sigh	75	1.3	4.0	0.0	2.7	6.7
Singing	523	7.5	3.8	1.9	19.5	22.9
Sink_(filling_or_washing)	292	3.4	4.8	0.0	29.1	31.5
Siren	77	11.7	15.6	2.6	33.8	41.6
Skateboard	83	1.2	3.6	0.0	39.8	39.8
Slam	351	2.3	9.4	0.0	31.1	35.9
Sliding_door	196	3.1	2.6	0.5	11.7	13.8
Snare_drum	741	1.9	3.0	0.5	8.1	9.4
Sneeze	64	3.1	23.4	0.0	14.1	35.9
Speech	1469	3.9	3.1	3.9	31.1	34.0
Speech_synthesizer	59	16.9	5.1	15.3	27.1	35.6
Splash_and_splatter	374	1.1	4.3	0.0	27.8	29.9

Squeak	389	4.9	3.3	1.3	26.5	30.1
Stream	214	7.5	1.4	0.5	21.0	23.4
Strum	176	11.4	2.8	0.0	33.0	35.8
Subway_and_metro_and_underground	305	13.8	17.0	0.0	21.3	33.8
Tabla	52	3.8	3.8	0.0	61.5	65.4
Tambourine	229	0.4	1.3	0.0	6.1	6.1
Tap	248	2.4	7.7	0.8	45.2	47.6
Tearing	276	3.3	3.3	0.4	52.9	54.7
Telephone	520	17.9	6.0	16.2	38.5	46.3
Thump_and_thud	299	2.7	17.4	0.3	32.1	42.1
Thunder	455	11.4	11.2	0.0	12.5	24.8
Thunderstorm	469	11.7	11.5	0.0	12.8	25.4
Tick	52	0.0	5.8	5.8	28.8	32.7
Tick-tock	96	0.0	2.1	2.1	31.3	33.3
Toilet_flush	206	1.5	10.2	0.0	18.4	25.7
Tools	788	2.2	6.6	0.8	34.6	37.7
Traffic_noise_and_roadway_noise	152	5.9	3.9	0.0	19.7	23.0
Train	338	7.4	11.8	0.3	16.0	25.1
Trickle_and_dribble	150	6.0	1.3	0.0	53.3	54.0
Truck	110	6.4	4.5	1.8	24.5	26.4
Trumpet	581	0.2	0.0	0.0	6.2	6.2
Typewriter	51	0.0	0.0	0.0	41.2	41.2
Typing	284	2.8	6.0	1.4	57.4	59.2
Vehicle	2680	8.2	7.2	0.4	21.5	27.5
Vehicle_horn_and_car_horn_and_honking	115	3.5	4.3	0.0	15.7	20.0
Walk_and_footsteps	377	7.7	4.8	0.3	32.4	35.3
Water	1360	5.5	3.3	0.3	24.9	28.5
Water_tap_and_faucet	280	2.1	1.4	0.4	27.1	27.5
Waves_and_surf	167	4.2	3.0	0.0	10.2	13.8
Whispering	170	3.5	1.8	9.4	33.5	38.8
Whoosh_and_swoosh_and_swish	272	1.8	2.2	0.0	7.0	9.2
Wild_animals	1641	4.3	2.3	1.5	13.7	17.4
Wind	294	12.6	6.5	0.3	18.7	24.8
Wind_chime	66	4.5	1.5	0.0	31.8	31.8
Wind_instrument_and_woodwind_instrument	2548	0.5	0.4	0.0	5.2	5.9
Wood	285	1.1	12.6	0.7	41.1	45.3
Writing	255	5.5	1.2	0.8	42.0	42.4
Yell	139	2.9	5.0	0.0	19.4	23.0
Zipper_(clothing)	291	10.0	2.1	1.0	45.4	47.1