

Overview of Honors Capstone Research

Mel Mark

Park University

Spring 2020

Table of Contents

I.	Introduction	3
II.	Gathering of Information and Literature	4
III.	Processing and Selecting Appropriate Methodologies	5-7
IV.	Metadata Extraction.....	8-9
V.	Conversion.....	9-10
VI.	Data Refinement.....	10-11
VII.	Training and Applying Models.....	11-15
VIII.	Looking Forward.....	15-16

INTRODUCTION

In the fall semester of 2019, the implementation phase of this honors research project at Park University with advisor Professor Joe Wang was started. This research was inspired by a love for music and teaching machines to predict qualitative data. Specifically, there was an interest in assigning human emotions to music from all over the world with the help of machine learning. This was not a small task, and required lots of research and collaboration with researchers all over the world, including Switzerland, California, Poland, Spain, and New Zealand (Image 1). This semester could be mainly divided into five sections respectively: gathering of information and literature, processing and selecting appropriate methodologies, metadata extraction, conversion, and analysis.

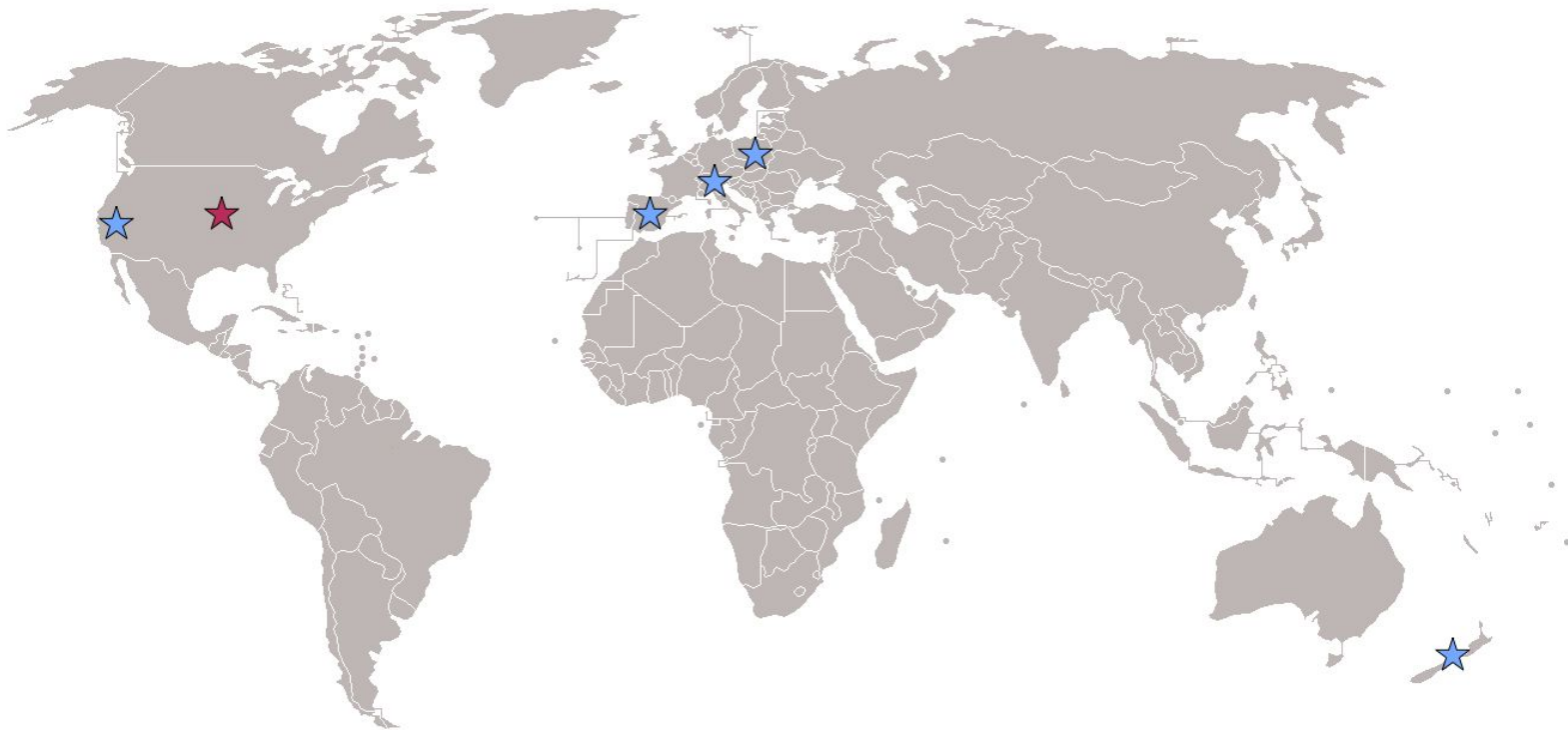


Image 1: Blue stars denote locations of some researchers that were collaborated with, the maroon star is where Park University is located in Parkville, Missouri.

GATHERING OF INFORMATION AND LITERATURE

As any researcher knows, the first step to any problem is to gather relevant information and scholarly literature to aid in one's journey. This started with browsing music information retrieval literature very broadly, looking carefully at datasets, intentions, and paths different researchers had taken. Then, as the search began to narrow, the focus shifted to solely machine learning papers that specifically had to do with music and emotions. This proved to be difficult as it is a relatively new field that does not have a lot of substance in its libraries.

As the research became deeper, it became apparent that an understanding vital to this field was missing: machine learning. As such, familiarization of concepts and terms concerning machine learning began by taking a course on Coursera and watching YouTube tutorials. This did help further an understanding of some of the fundamental theories and practices behind machine learning, but it was difficult to fully grasp a lot of it without tinkering with it. As a result, the long journey of sifting through all the different methods and approaches various researchers before me had taken began, searching for the one that would best work for this problem and that would allow new knowledge to be put to the test. Eventually, the New Zealand tool Weka was settled upon. Weka reads ARFF files, a file type only used for Weka, and is able to preprocess data, classify data, cluster data, associate data, and visualize data. This tool was specifically liked because it had a GUI as opposed to just a command line tool. It is also widely used across the field of machine learning and has reputable and accredited documentation and research.

PROCESSING AND SELECTING APPROPRIATE METHODOLOGIES

While the literature regarding emotive machine learning with music information retrieval is not abundant, there is a great variety in the approaches that are reported. Because of this, it took a lot of reading and searching to find which methods supposedly had the lowest margin of error and best datasets. Between the different regression algorithms implemented to the myriad of differences among datasets, it was overwhelming to choose one way or another when there is more than one way to bake a cake. Two papers that heavily assisted in picking methods are “Detecting Emotion in Music” by Li and Ogihara and “Music Emotion Recognition: A State of the Art Review” from Ithaca College and Drexel University. Both papers detailed methods researchers experimented with before and included the results and accuracy of each.

Another field that needed more preparation was psychology, especially regarding the various models used to plot mood. As such, reading on the many methods of enumerating mood and emotions on both two dimensional and three dimensional graphs was necessary. Eventually, the model known as Thayer’s Model was chosen to two dimensionally plot emotions into four categories (Thayer, 2000). The two axes represent valence and arousal, two measurements that ultimately determine which quadrant and which emotion. The four quadrants were further divided into nine subsections each, with three emotions representing three subsections in each quadrant. While there are 36 subsections in 4 quadrants, there are only 12 moods represented (pleased, happy, excited, annoyed, angry, nervous, sad, bored, sleepy, relaxed, peaceful, calm) because 36 divided by the 3 subsections occupied by each mood equals 12.

Now that a model to plot emotions existed, it was then needed to find an appropriate dataset that had a good number of songs of the same length and annotated with valence and

arousal values. This was vital to build the training set of data for the machine learning algorithms. This was a tall order to fill as this field is already so niche, there are not many datasets created, let alone public. After lots of digging, eventually the DEAM: MediaEval Database for Emotional Analysis in Music (Soleymani, 2016) was discovered. It was created by researchers in Switzerland and contained lots of information that was not relevant to this research. However, it did contain valence and arousal values for 1,745 songs and their 45 second audio files. It was not clear originally how they extracted the arousal and valence, so contact was attempted. After searching the Internet for the researcher's name, his personal website popped up which stated he was now a professor in California. An email was quickly sent to him and he immediately got back stating that they had not extracted arousal and valence directly like originally thought, instead, they used turkers to manually extract the values. Of course, this is not ideal as it is subjective and not replicable. However, it was better than nothing.

The next step was plotting all the songs in the dataset on the two dimensional model by Thayer. To do this, the origin of the graph had to be found, which was not (0, 0) as one might expect. Instead, the mean of each was found and was made the origin (4.9, 4.8). Then the minimum and maximum values of the arousal mean column and the valence mean column were subtracted to find the differences. Once the differences were obtained, each was divided by 6 (three subsections high and wide in each quadrant, times 2 equals 6) to find the increment. The boundaries were drawn for each subsection (Figure 1) and a program in Python3.7 was wrote to automatically take in the valence and arousal mean columns and calculate and write each corresponding mood in a new CSV. Each subsection in the map essentially represents that area of mood based on the two dimensional plotting of arousal and valence. The subsection for the

song is determined based on where it lands on the graph and quadrant. Each mood has three subsections that neighbor each other so as to divide emotionally similar songs. Now that it was learned that they manually extracted the valence and arousal and the dataset was annotated with moods according to Thayer's model, the next objective was to automatically extract the arousal and valence, which is much easier said than done.

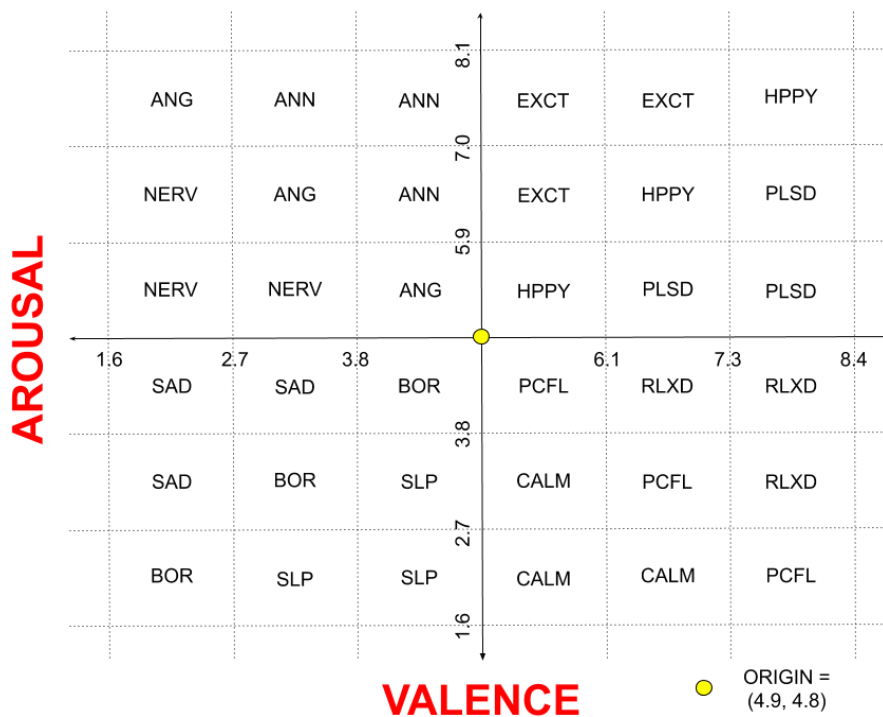
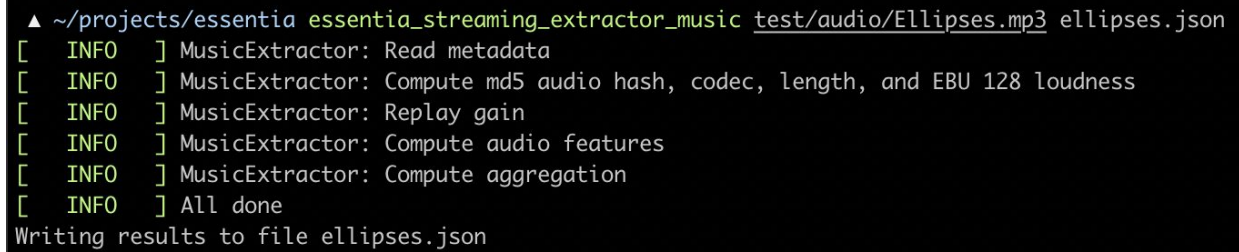


Figure 1: Hand-mapping each quadrant, section, and subsection for DEAM.

METADATA EXTRACTION

At the beginning of trying to extract music data, it was naively thought that arousal and valence could be directly extracted from songs. Instead, what was learned is that arousal and valence are psychological terms that mean nothing to music. In order to calculate arousal and valence, calculations would have to be made from other aspects of music data. Of course, this was not common knowledge and reviewing scholarly literature was necessary once again. Eventually a gem was found, a Polish researcher named Jacek Grekow published a paper detailing differences in music metadata extractors and the pros and cons of each (Grekow, 2018). The two extractors he focused on were Marsyas and Essentia. Marsyas is a Greek command line tool written in C++ and Essentia is a Spanish command line tool written also in C++. Both extracted metadata from music, but Grekow noted that Essentia is more detailed and provides more features, which would ultimately lead to greater accuracy in detection of emotions. Because of this and technical complications in using Marsyas, Essentia was chosen.

Essentia has three high-level categories it puts its features in: low-level, rhythm, and tonal. It takes the audio file and quickly and efficiently outputs the data in a JSON or YAML file (Image 2). In addition, Essentia calculates thorough statistics for each feature extracted, including mean, geometric mean, power mean, median of an array, all its moments up to the fifth order, energy, and the root mean square (RMS). The only issue with Essentia is that it outputted the data in only JSON or YAML, with no option for ARFF files, which are the only kind Weka can read. As a result, the files would need to be converted somehow, which would prove a challenge.

A terminal window with a black background and green and white text. The prompt is a green triangle followed by a path: ~/projects/essentia. The command is essentia_streaming_extractor_music test/audio/Ellipses.mp3 ellipses.json. The output consists of several INFO messages in brackets, followed by a final message indicating the results are written to a file.

```
▲ ~/projects/essentia essentia_streaming_extractor_music test/audio/Ellipses.mp3 ellipses.json
[ INFO ] MusicExtractor: Read metadata
[ INFO ] MusicExtractor: Compute md5 audio hash, codec, length, and EBU 128 loudness
[ INFO ] MusicExtractor: Replay gain
[ INFO ] MusicExtractor: Compute audio features
[ INFO ] MusicExtractor: Compute aggregation
[ INFO ] All done
Writing results to file ellipses.json
```

Image 2: *Essentia reading and computing data within seconds.*

CONVERSION

Because ARFF files are only for Weka, there are not any converters publicly available. Weka does have a built in ‘converter’ tool but it only works with very specific file types with very specific data. After conversing with some researchers who developed Weka, it was clear that a custom built tool was needed. Luckily both the input and the output types were not too complicated, so building a converter for the first time would not be totally out of the question. ARFF files basically have a set of labels or ‘attributes’ that is paired to a data value, depending on position within a comma separated list. There were a wide variety of languages to choose from to write this program but ultimately chose Python3.7 for its simplicity and robust documentation.

The first challenge was extracting just the numbers of the JSON file Essentia outputted. The research was not interested in the labels, just the numerical values and two non-numerical values (key signature and major/minor key). This was made even more difficult by the fact JSON files from Essentia contain multiple levels, and it is not possible to grab just the values for this to work. There was another challenge within this: some labels contained numbers, such as ‘dmean2’ or ‘melbands128’, so there had to be a system to somehow figure out how to

differentiate numbers in labels and actual numerical values. Also, a lot of numbers contained e+ or e- to denote scientific notation, so in addition to some labels containing numbers, some numbers contained letters. This was circumvented by a few conditional statements checking for those exceptions listed above. Some temporary lists and strings as placeholders were also used to manipulate data before assigning the finalized data to the output variables.

Now that just the numbers were able to be extracted, the actual converter that took the JSON data and output it to an ARFF file had to be written. In Essentia, all labels are the same for each song except one. There is a label called 'beats_position' that can vary with how many times it appears. Sometimes there are over 150 beats_position numbers, but sometimes there are under 75. A loop was able to be written that detected how many beats_position labels to generate in the ARFF files that solved that problem. Finally, Weka was able to open up successfully converted JSON → ARFF files and analyze them. Now all that was left was to clean up the dataset acquired previously.

DATA REFINEMENT

Before the analysis and classification of data could begin, the dataset needed to be of the highest quality, so that the predictions would be as accurate as possible. Some of the moods the program assigned to songs were a little questionable. To avoid outliers or inaccurate data, the process of going through and listening to each 1,745 45 second audio clip began. Each one that was found to be accurate was marked and added to an improved dataset. Image 3 shows the original 1,745 dataset along with annotations of whether the predictions were accurate or not. The predictions annotated as accurate were then put in the final, refined dataset. Once this was done, the predicted emotions in songs in Weka was more accurate.

what was wanted based on Thayer's model. The 16 critical features he defines are segmented into three categories based on Essentia's output: lowlevel, rhythm, and tonal. The 10 lowlevel critical features are melbands crest and kurtosis, spectral energy, entropy, flux, rolloff, and skewness, zero crossing rate, gfcc, and mfcc. The 3 rhythm critical features are danceability, onset rate, and beats loudness band ratio. The remaining 3 tonal critical features are chords strength, hpcp entropy, and key strength.

Based on these critical features, the data was ready for preprocessing. Originally there were 4,377 features outputted by Essentia for each of the 120 MP3s, so there was a big need for a way to quickly select thousands of features and remove them to slim down the data. Weka integrates PERL RegEx (Regular Expression) as a way to pattern-match features and manipulate them. Although lacking experience with PERL RegEx, it was quickly learned and was able to trim it down quite sizably (Image 4).

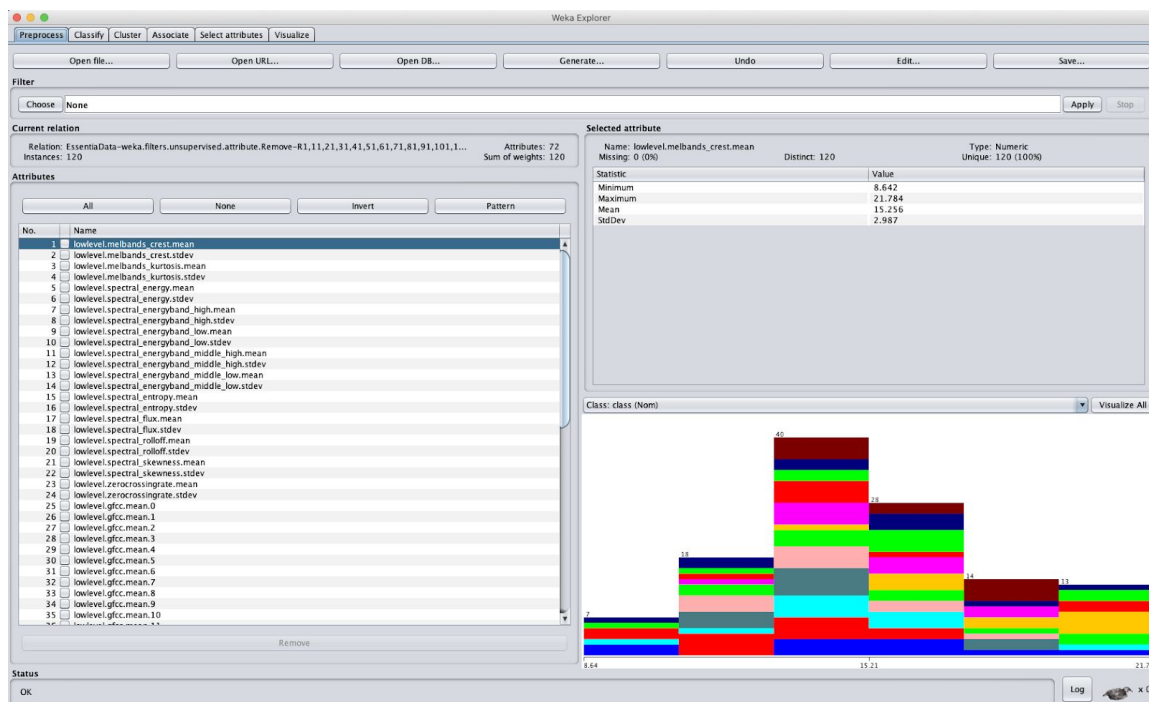


Image 4: *Cherry picked features preprocessed with PERL RegEx in Weka.*

Based on the initial literature gathering, there were specific machine learning algorithms that were desired that would best match the specific problem. Of the dozens of algorithms, 4 were settled upon: sequential minimal optimization (SMO), neural network (what Weka calls multilayer perception), simple logistic, and J48 tree. The SMO algorithm is an iterative algorithm designed to solve the optimization problem regarding binary classification. It does this by breaking the problem into the smallest possible sub-problems before being analytically solved. Basically, it has to find a minimum of a one-dimensional quadratic function using Lagrange multipliers that meet the Karush–Kuhn–Tucker (KKT) conditions. Neural networks are loosely modeled off biological brains in its abilities to pick up new patterns without being explicitly taught. The simple logistic algorithm presumes that the connection between the measurement variable and the natural log (\ln) of the odds ratio is linear. Lastly, the J48 tree has the structure of a flowchart with nodes and internodes. Decision trees, like J48 trees, were more popular about ten years ago but still used today. Some of these algorithms are far more complicated than others, and all operate very differently, but they all predict fairly similar results most of the time. Each of the 120 songs' data was applied to each algorithm and generated 4 models (Image 5). Finally, models could be applied to new songs and output predictions.

=== Evaluation on training set ===

Time taken to test model on training data: 0.03 seconds

=== Summary ===

Correctly Classified Instances	112	93.3333 %
Incorrectly Classified Instances	8	6.6667 %
Kappa statistic	0.9273	
Mean absolute error	0.0621	
Root mean squared error	0.1404	
Relative absolute error	40.668 %	
Root relative squared error	50.8165 %	
Total Number of Instances	120	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1.000	0.009	0.909	1.000	0.952	0.949	0.999	0.991	pleased
	0.900	0.009	0.900	0.900	0.900	0.891	0.999	0.991	happy
	0.800	0.000	1.000	0.800	0.889	0.886	0.980	0.893	excited
	1.000	0.009	0.909	1.000	0.952	0.949	0.995	0.943	annoyed
	0.800	0.009	0.889	0.800	0.842	0.830	0.992	0.933	angry
	1.000	0.018	0.833	1.000	0.909	0.905	0.998	0.981	nervous
	0.900	0.000	1.000	0.900	0.947	0.944	1.000	1.000	sad
	1.000	0.018	0.833	1.000	0.909	0.905	0.995	0.954	bored
	1.000	0.000	1.000	1.000	1.000	1.000	0.997	0.977	sleepy
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	relaxed
	0.900	0.000	1.000	0.900	0.947	0.944	0.994	0.953	peaceful
	0.900	0.000	1.000	0.900	0.947	0.944	0.998	0.983	calm
Weighted Avg.	0.933	0.006	0.939	0.933	0.933	0.929	0.996	0.967	

=== Confusion Matrix ===

	a	b	c	d	e	f	g	h	i	j	k	l	
10	0	0	0	0	0	0	0	0	0	0	0	0	<-- classified as
1	9	0	0	0	0	0	0	0	0	0	0	0	a = pleased
0	0	8	0	1	1	0	0	0	0	0	0	0	b = happy
0	0	0	10	0	0	0	0	0	0	0	0	0	c = excited
0	0	0	1	8	0	0	1	0	0	0	0	0	d = annoyed
0	0	0	0	0	10	0	0	0	0	0	0	0	e = angry
0	0	0	0	0	0	9	1	0	0	0	0	0	f = nervous
0	0	0	0	0	0	0	10	0	0	0	0	0	g = sad
0	0	0	0	0	0	0	0	10	0	0	0	0	h = bored
0	0	0	0	0	0	0	0	0	10	0	0	0	i = sleepy
0	0	0	0	0	1	0	0	0	0	9	0	0	j = relaxed
0	1	0	0	0	0	0	0	0	0	0	9	0	k = peaceful
												9	l = calm

Image 5: Example classification using the SimpleLogistic algorithm.

A few songs were chosen to initially analyze against the models, and the models' predictions were agreeable. A popular song at the time was *Juice* by Lizzo, also known as Melissa Jefferson. The song was divided into 9 sections: intro, first verse, first pre-chorus, first

chorus, second verse, second pre-chorus, second chorus, bridge, and last chorus respectively.

After segmentation each were analyzed against the models in addition to the entire 3 minute song as a whole (Figure 2). What was interesting was that the sections that repeated, like pre-choruses and choruses, had pretty much the same results, with the same levels of confidence. The song ranked ‘pleased’ across the board with the exception of the second verse, which was predicted as 64% ‘angry’ and 36% ‘happy’. The second verse of the song is when the mood shifts a bit as Lizzo gets more sassy, and it is noticeable when listened to. Therefore, the models’ results with this song were satisfactory, as it was able to accurately map the mood of the song over time.

type of ML	juice_o g	juice_1 _intro	juice_2 _verse1	juice_3 _precho rus	juice_4 _chorus	juice_5 _verse2	juice_6 _precho rus	juice_7 _chorus	juice_8 _bridge	juice_9 _chorus
SMO	pleased, .167	pleased, .167	pleased, .167	pleased, .167	pleased, .167	happy, .167	pleased, .167	pleased, .167	pleased, .167	pleased, .152
simple logistic	pleased, .821	pleased, .972	pleased, .969	pleased, .664	pleased, .737	happy .426	pleased, .341	pleased, .678	pleased, .976	pleased, .311
neural network	pleased, .962	pleased, .541	pleased .845	pleased .739	pleased, .777	angry .406	pleased, .718	pleased, .81	pleased, .978	pleased, .64
J48	angry, .667	happy, 1	excited .667	pleased, 1	angry .667	angry .667	pleased, 1	angry .667	angry .667	angry .667
result with weights	75% pleased	63% pleased	75% pleased	100% pleased	72% pleased	36% happy	100% pleased	71% pleased	76% pleased	62% pleased
	25% angry	37% happy	25% excited		28% angry	64% angry		29% angry	24% angry	38% angry

Figure 2: *Juice* by Lizzo segmented based on song structure and analyzed by each model.

LOOKING FORWARD

Ideally, the research would like to further segment more songs of a greater diversity to make more predictions over time. It would also like to explore more ways to tweak the models to make them more accurate. Another goal would like to do is automate the process. It currently takes hours to take the MP3, trim it, run it through Essentia to generate a JSON, put the JSON file through the Python converter to generate an ARFF, then preprocess the ARFF in Weka using Perl RegEx, apply it to each model, and record the results.

There are a wide variety of applications for music, and therefore lots of applications for this research. It would satisfy the authors to apply this research to something that would benefit the community, however that may be.

References

- Grekow, J. (2018). Audio features dedicated to the detection and tracking of arousal and valence in musical compositions. *Journal of Information and Telecommunication*, 2(3), 322-333.
- Jefferson, M. (2019). Juice [Recorded by Lizzo]. On Cuz I Love You [MP3]. New York City, NY: Atlantic Records. (2018 December)
- Kim, Y. E., Schmidt, E. M., Migneco, R., Morton, B. G., Richardson, P., Scott, J., ... & Turnbull, D. (2010, August). Music emotion recognition: A state of the art review. In *Proc. ISMIR* (Vol. 86, pp. 937-952).
- Li, T., & Ogihara, M. (2003). Detecting emotion in music.
- Soleymani, M., Aljanaki, A., & Yang, Y. H. (2016). DEAM: MediaEval Database for Emotional Analysis in Music.
- Thayer, J. F., & Lane, R. D. (2000). A model of neurovisceral integration in emotion regulation and dysregulation. *Journal of affective disorders*, 61(3), 201-216.