Language Identification of Vocal Lyrics in Songs

Russell Brown RussellRok@gmail.com

Taylor LeBlond wtleblond@gmail.com

Christopher Lam chris_lam25@hotmail.com

Abstract

Automatic language identification (LID) has been a challenging research topic for many studies over the course of several years. By looking at certain features of different languages, this study makes an attempt to derive a method for an accurate language detection of lyrics in songs. By isolating the vocal track from the background music, this program analyzes the extracted vocals by looking at the occurrence of sharp instances of frequencies over time. Through the exploration of multiple proposed classification methods, each gave their own outcome and proved efficient in their own way.

1. INTRODUCTION

The goal of this project was to create a program to determine which language a song is being sung in. The song will be played through a microphone or loaded in as a file and the program will then classify the song by language, with a database of 360 songs as the training set. Multiple classification methods were used and compared against each other.

A few studies have attempted a similar task. Chandrasekhar et al. [1] achieved a 47.8% accuracy in classifying songs based off of audio features such as Mel-frequency cepstral coefficients (MFCCs) and stabilized auditory images (SAI), as well as music video features, from a database of 25000 songs in 25 different languages. Tsai and Wang [2] achieved an 80% accuracy in classifying songs using grammatical modeling from a database of 64 songs, 32 in English and 32 in Mandarin. Mehrabani and Hansen [3] achieved an 82.7% accuracy using Parallel Phone Recognition followed by Language Modelling (PPRLM) combined with prosodic LID on approximately 12 hours of singing data in Farsi, Hindi and Mandarin. Schwenninger et al. [4] achieved a 64% accuracy using various LID techniques on 205 pop and rock songs in English or German.

The final outcome was to have the program obtain an accuracy above 40% in differentiating the four languages: English, French, German and Japanese. Three of the studies mentioned above used fewer languages than what was proposed for this project, so a higher degree of error was expected, and thus, the expected accuracy was lower for this project. While the Chandrasekhar study received a reasonable accuracy for classifying amongst a far greater range of languages, the accuracies were very low for English, French and German, which were three of the four languages tested in this study.

Multiple methods have been explored to determine which method would produce the highest classification accuracy. Individual classifications methods will not be used for each individual language, but a single method will be used for all languages for the program to differentiate between the languages.

The worst case scenario for the program would be if the classification algorithm does not reach the 40% range. The minimum viable product would be a classification algorithm that has an accuracy just above 25%. If the classification for the audio is below 25%, the experiment will be considered a failure as random guessing would become a more viable option at that point.

2. METHODS OF ANALYSIS

As the features being extracted are exclusively from vocals, the vocal part of each track was extracted and used as opposed to the unedited tracks. This was done as it was expected that the background music of a track, especially percussive sounds, would cause interference with the results. This was accomplished using the Spleeter program [5]. Librosa [6] was used to load the extracted vocal tracks into the Python programs used for classification. The use of unedited tracks will be used for comparison.

3. DATABASE INFORMATION

The database consisted of 360 .wav files of 30-second cuts of songs in English, French, Japanese and German (90 for each language). The music used was an assortment of tracks from top playlists and Youtube playlists, with a relative mix of genres. Attempts were made to prevent certain genres from being biased towards a language, although this wasn't strongly enforced.

The .wav files were cut to 30 seconds to ensure song length was not a factor in the end results, and to speed up the process. Each cut starts 35 seconds into the song, as it was observed that many, if not most, of the tracks had instrumental introductions devoid of any vocals. It should be noted that specifically choosing which part of a song to cut for analyzing is subjective, as intentionally choosing the part of the song that specifically produces the "most" of a result is essentially

moving the goalposts. For this reason, it was expected that there would be outlier cases that would be nearly impossible to accurately predict.

4. TREATMENT OF DATA

4.1 Vocal extraction

The vocal extraction tool used for this project was Spleeter, a source separation library made by Deezer. Although Spleeter has the ability to separate the audio into many different stems, the only stem required was the vocal stem and therefore the 2-stem separation method worked for this project. The Python code calls Spleeter through the command line.

4.2 Sampling

Using the microphone, a song is recorded for approximately 30 seconds. The program then extracts the vocals, analyzes the recording, and compares it to the database for the nearest matching language.

5. TOOLS AND RESOURCES

GitHub was used to share the project and provide easy access to the code and data for the project members. Some of the vocal extractions were listened to to ensure Spleeter was functioning properly. Data was stored as .wav files for ease of access and storage. As expected for most Python programs, the NumPy library [7] was heavily used for arrays and mathematics. The SciPy library was used for filtering signals [8]. The SciKit-Learn (or Sklearn) library was used for machine learning in one of the classification methods [9]. As mentioned previously, some tools from Librosa were used for various tasks.

6. PRELIMINARY TESTS

Initially, a vocal extraction method using modified code from Librosa was used for extracting the vocals from tracks. It did not produce ideal results and was abandoned when Spleeter was found to produce far greater quality extractions. However, Librosa was still used to cut the files and input them into the program.

The first attempts as classification were to test some standard features used in MIR. The first features tested were the mean and standard deviation of the spectral centroid of each unedited track (*Figure 1*), followed by those of each vocal extraction (*Figure 2*), and the mean and standard deviation of the Mel-frequency cepstral coefficients (MFCCs) of the vocal extractions (*Figure 3*). Overall classification accuracy was below or well below random guessing range (<25%) for each case, so all of these methods were abandoned.

It should be noted that these methods were more or less used for testing classification, and it was not expected that they would necessarily work immediately, if at all.

In the following diagrams, the data points plotted all represent variations of the mean versus the standard deviations following different variables. The blue points represent songs in English, orange for French, green for Japanese, and red for German.

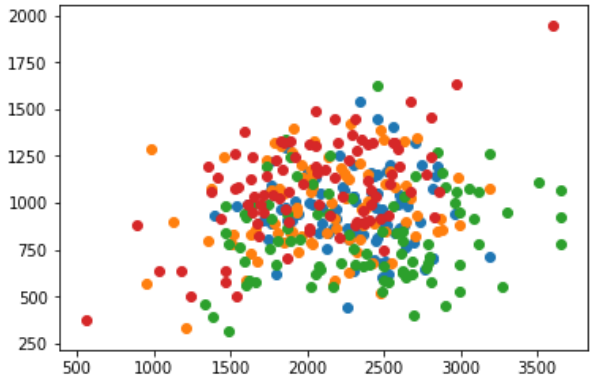

Figure 1: *Scatter plot of the mean versus standard deviation of the spectral centroid of the original tracks.*

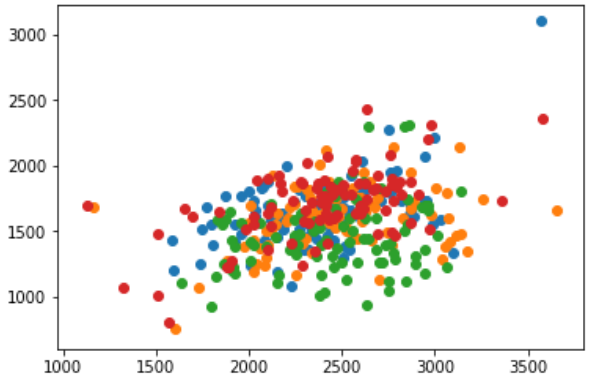

Figure 2: *Scatter plot of the mean versus standard deviation of the spectral centroid of the same tracks run through Librosa's vocal extractor.*

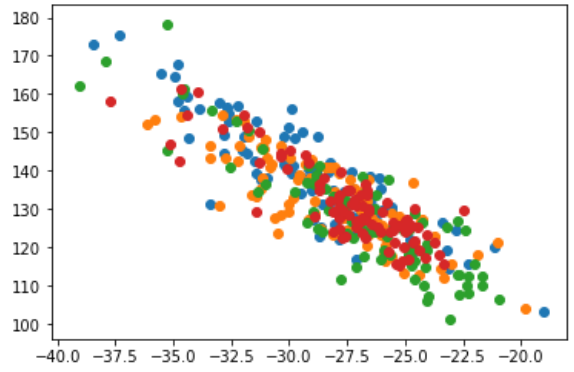

Figure 3: *Scatter plot of the mean versus standard deviation of the MFCCs of the same tracks run through Librosa's vocal extractor.*

The diagrams above do not show a great representation of the separation of language. As observed in each diagram, the regions for each language largely overlap each other. This made it near impossible to determine the language classification with spectral centroids and MFCCs alone. Therefore, the next step of action was to research more parameters for comparison to improve the accuracy of the classification algorithm.

7. CREATING THE DATA

Librosa was used to read in each file, and cut the length down to 30 seconds, starting 35 seconds into each track. Spleeter, a program that uses pretrained models to separate sound sources from one another [5], was used to extract the vocals into separate .wav files. Many other studies used vocal extraction methods [1,2,3] but all are older than Spleeter, which is a relatively new piece of technology.

It was observed that voiceless fricative consonants (e.g. “s”, “sh”, “th”) occur at much higher frequencies than other consonants [10] (*Figure 4*). Therefore, an extraction method was developed based off of the notion that certain sibilants at certain frequencies may occur more often in one language compared to another.

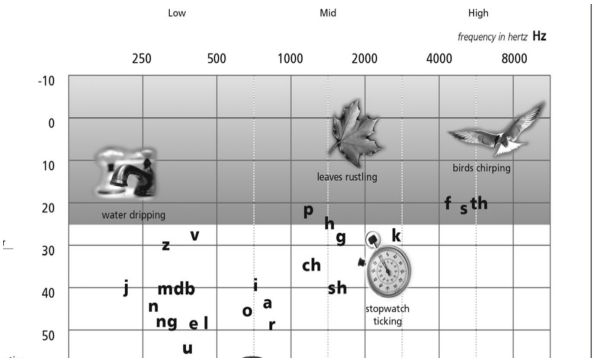

Figure 4: *A diagram of rough estimates for the frequency of certain consonant sounds. Source: [10].*

Instead of trying to extract the number of instances a recognizable consonant (such as “s”) occured in a track, the number of instances sharp consonance appeared within various ranges of frequencies was used. Each vocal extraction was run through twelve different bandpass filters, each being 1000 Hz in range, with the lowest having a low threshold of 3000 Hz and each separated by 1000 Hz, and one highpass filter with a threshold of 15000 Hz, as follows (all filters were 10th-order):

3000 Hz - 4000 Hz
4000 Hz - 5000 Hz
…
14000 Hz - 15000 Hz
15000 Hz + (highpass filter)

This filtering results in instances in a signal outside the frequency range for each filter having a significantly reduced magnitude, causing frequencies within the range to heavily stand out. This allowed the program to pick out sharp instances of consonance within each range, which were represented by peaks. The number of peaks within the filtered signal (a peak being a sample above 0.375 times the maximum amplitude, and at least 4000 samples away from the previous counting peak) was counted and placed in an array with a length of 13, each slot corresponding to the number of peaks for each filter.

After all the occurrences were counted, it was possible to plot the average number of occurrences of each language per filter onto a graph (*Figure 5*). This was very important because it demonstrated a clear difference between the four languages that could be used for classification. For example, German (red), had a relatively high occurrence of consonance for the six highest filters. Conversely, Japanese (green), had a relatively low occurrence of consonance for the same filters, but a high occurrence in the fifth and sixth filters.

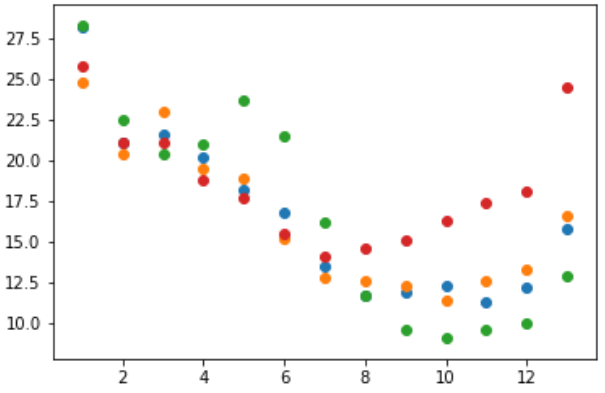

Figure 5: *Scatter plot for the average occurrences of frequencies for each language in 360 songs. English is Blue, French is Orange, German is Red and Japanese is Green.*

Unfortunately, the English and French averages were uniformly close to each other and were expectedly the hardest for the classifiers to predict accurately.

There are a few important components to keep in mind when conducting speech analysis, many of which can be applied to language identification in songs. Sreenivasa Rao et al. [11] had a list of requirements that the test database tried to capture. The computation time should be short, which was done by shortening the songs. The second characteristic was no language bias. This was harder to achieve, but the range of different music genres from different years attempted to balance this. The system should be able to tolerate different volumes of music with no issue and the system should be able to add languages easily [11].

8. COMPARING AND ANALYSIS METHODS

8.1 Method 1

The first method was a custom method that didn't use any machine learning libraries. This method looked at certain bandpass filters and based on the results of those filters made assumptions about the data. *Figure 5* showed the greatest distances between points (not including the highpass filter) occuring in the 10th, 11th and 12th bandpass filters (12000-15000 Hz). The data was then compared against every song in the database, but only with these three bands. This was done by comparing the absolute distance of each band across each language to determine which language is the most likely to occur. This method used a decision tree-like format in order to get a result.

The first decision determined if the language would be divided into one of two categories, English/French or German/Japanese. Beyond this, the decision tree made certain values in certain bands worth more. For example, if German has a value of x>=50, the German points will get a +3 added on to their weight. Similar weights are applied to various other qualities across both sides of the decision tree.

Using this method resulted in an English accuracy of 28.9%, a French accuracy of 20.0%, a Japanese accuracy of 42.2%, and a German accuracy of 46.7%. The total accuracy was therefore 34.4%. (*Figure 6*)

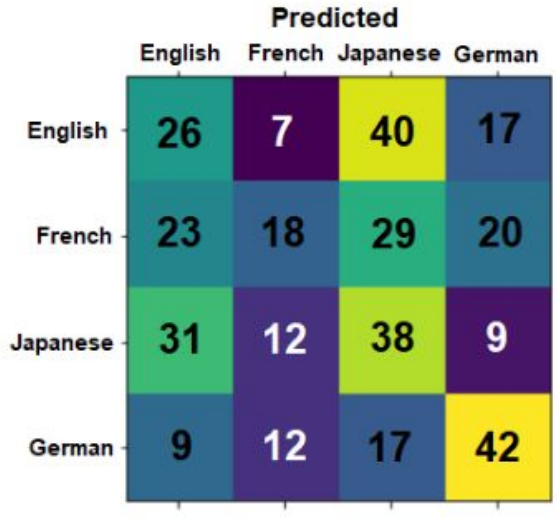

Figure 6: *The number of times, out of 90, a song was classified using Method 1, correct language to guessed language. For example, English was classified as English 26 out of 90 times, but was classified as French 7 out of 90.*

8.2 Method 2

A second custom method using a different strategy was also attempted for classification while also not using any machine learning libraries. The numbers of occurrences in a specific filter were placed into arrays corresponding to language. These arrays were then sorted from lowest to highest. The case for the highpass filter is graphed out below. (*Figure 7*)

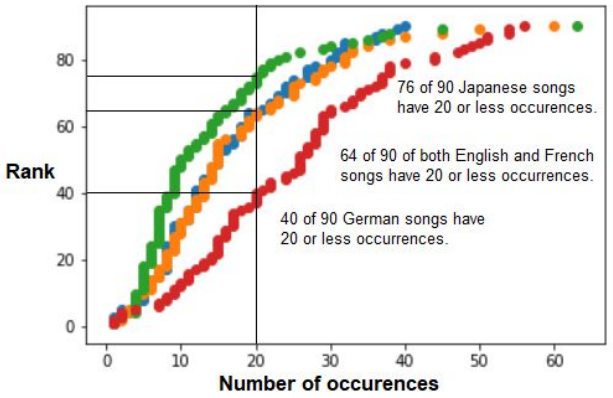

Figure 7: *This graph demonstrates that the vast majority of Japanese songs, but only a minority of German songs, have less than 20 occurrences for the highpass filter. English is Blue, French is Orange, German is Red and Japanese is Green.*

The array containing the number of occurrences within each filter for a song is input and each number is ranked against the results for the respective filter for each language. For each language, the following score is added for each filter:

$$\left|\frac{rank-(90-rank)}{2}\right|$$

The language with the *lowest* cumulative score is the predicted language. Essentially, scores are given by how many ranks away from the average the input song is. This means that an extreme outlying case in a filter results in an equal or near-equal score for each language, which has a negligible overall effect. In contrast to the "winner-take-all" scoring of Method 1, each language receives a score for each filter in this method.

Using this method resulted in an English accuracy of 35.6%, a French accuracy of 16.7%, a Japanese accuracy of 51.1%, and a German accuracy of 56.7%. The total accuracy was therefore 40.0% (*Figure 8*).

Overall, this is an improvement over Method 1, and it was able to correctly predict a *majority* of the songs in two languages, Japanese and German. However, its accuracy for French, which was already below random guessing range, decreased in comparison.

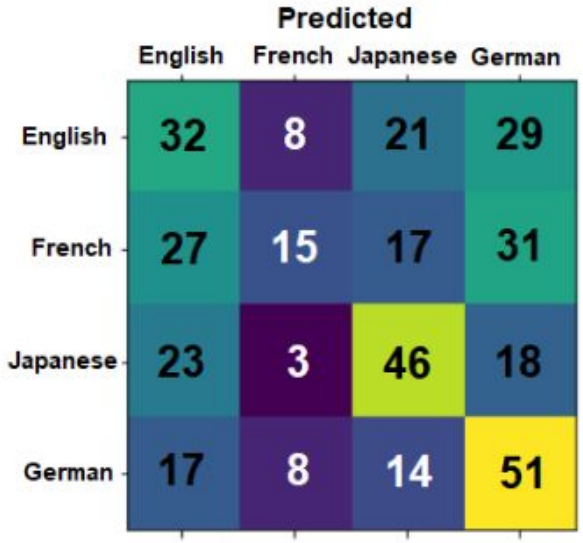

Figure 8: *The number of times, out of 90, a song was classified using Method 2, correct language to guessed language. For example, English was classified as English 32 out of 90 times, but was classified as French 8 out of 90.*

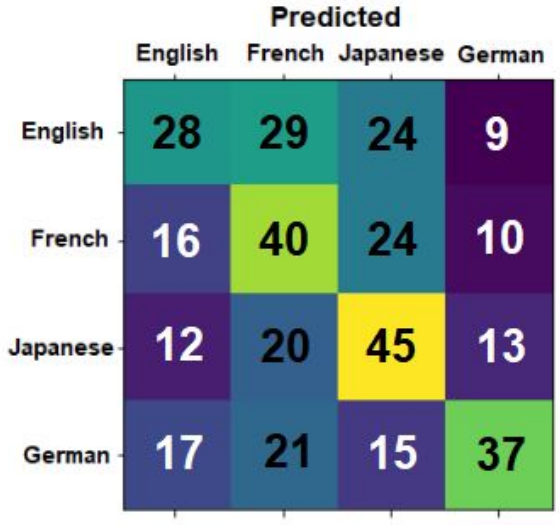

Figure 9: *The number of times, out of 90, a song was classified using Method 3, correct language to guessed language. For example, English was classified as English 28 out of 90 times, but was classified as French 29 out of 90.*

8.3 Method 3

The third method utilized SciKit-Learn, a Python library used for machine learning, for classification. The *metrics.accuracy_score* function was used for this. This method compared how similar two numpy arrays were from one another. Similar to previous methods, the song was compared against every song in the database. However, everytime it predicted a particular language, that number scored a point. If the 2 lowest languages were tied, both languages would score a point. After all 90 songs have been compared, the predicted language is then outputted.

Using this method resulted in an English accuracy of 31.1%, a French accuracy of 44.4%, a Japanese accuracy of 50.0%, and a German accuracy of 41.1%. The total accuracy was therefore 41.7%.

Overall, this method produced better results when compared with the previous two methods. Though this is largely due to the unexpectedly large increase in classification accuracy for French, which was very inaccurate for the other two. It was worse than both previous methods at predicting German and slightly worse at predicting English and Japanese than Method 2. Unsurprisingly, a slim plurality of the English songs were predicted as French (*Figure 9*).

8.4 Comparison of Methods

Overall the three analysis methods did an effective job at classifying the four languages. When looking at these methods, there was a clear trend that Japanese and German were the easiest to classify. This was to be expected based on observations of *Figure 5* as Japanese and German are individually differentiable. As expected from the same observations, and demonstrated by the classification results of Methods 1 and 2, English and French are far more challenging to classify. Conversely, French had a far higher classification rate in the Method 3. Though it should be noted that a plurality of English songs were classified as French, which once again demonstrated correlation between the two languages.

8.5 Comparison with original tracks

The three methods were tested on the unedited song files that still contained the background music. These methods did not yield desirable results due to the bandpass filter picking up peaks from the background instruments. This lead to an inaccurate number of peaks, many of which had nothing to do with language of the vocals. This gave insight on how the use of vocal separation was key for the goal of this project. The non-vocal extracted music didn't classify above 30% across the different languages. Non-vocal extracted music may still be viable when determining which language is being sung, but not with the methods used in this report.

8.6 Comparison of methods to language detection in spoken language

Although the phoneme detection in this algorithm was not a true phoneme detector, the idea

behind it was still present. Phoneme detection was used in other studies involving normal language recognition tasks [12,13]. Matějka [13] used speech feature extraction, a neural network, and language modeling in order to output a language. This was an effective way at differentiating the languages and would have been a valid option to explore in LID in song. However, languages do have more aspects than just phonemes to distinguish them apart. As tested by Mary and Yegnanarayana [14], prosodic features such as intonation, emotion and rhythm are important characteristics of languages. Due to the nature of language and speech in musical fashion, these were not explored as the type of songs have an influence on these factors in lyrics.

Similar to this study, previous studies also used 30-second samples [2,13,14,15,16,17,18]. Adami and Hermansky [13] attempted to segment utterances by inflection points for LID of normal speech. Martin and Pryzbocki [15] tested the accuracy of spoken language through several different methods. Overall, English had the lowest percentage of incorrect guesses, which is the opposite to the error results of this study. With all the different language detection methods, human perception is the best at distinguishing language as demonstrated by Navrátil [19], where the overall accuracy was 96% with spoken language. This study used four of the five same languages as Navrátil.

Many of these studies also use 10-second and 3-second clips. Though this was attempted during this project, the results were less accurate or offered little insight on the program's effectiveness that it was later discarded.

9. CONCLUSION

All methods had a higher overall classification accuracy than randomly guessing. Methods 2 and 3 both reached and surpassed the expected 40% accuracy rating, which means that the experiment was a success. Additionally, Method 2 was able to accurately classify the majority of songs in Japanese and German. Method 1 received the lowest accuracy with an overall classification of 34%. Method 2 had an overall classification of 40%, and Method 3 had an overall classification accuracy of 42%.

Whether it be for the quality of results or the efficiency in which it can analyze songs, there are many ways in which this project could have been and can be improved. The inclusion of more songs in each language to create the database would have aided in gaining a more accurate representation of frequency occurrence averages. If the database had a better average representation, the analysis of songs input into the program would potentially have had a higher accuracy in determining which language was present in its vocal source.

Though it could be a detrimental factor in creating the database, the spread of song genres was not as wide as hoped. Due to the time constraints of the project, the songs selected were not picked to equally balance the genre numbers and therefore perhaps skewed the bias of certain consonant sounds. An example of this is how the number of German rap songs strongly outweighed the number of Japanese rap songs. Perhaps a more careful selection of songs would have yielded a more balanced database for the languages.

The use of multiple classification techniques could have also been implemented into the program. Currently, the program is only counting the number of times certain frequencies are present in a song. It then proceeds to try and match it to a corresponding language in *Figure 5.* The inclusion of different key features of the selected languages would provide more variables in which the program would be able to use.

Including more languages is another option to be explored. Although it has not been discussed as to what languages would be or should have been added, the inclusion of other languages outside of English, French, German and Japanese would give a better view as to what parts of the program are working as expected and which ones are 'getting lucky'. It would provide more information about the program and how it is determining which language takes priority over which.

10. REFERENCES

[1] V. Chandrasekhar, M. Sargin, D. Ross: *Automatic Language Identification in Music Videos with Low Level Audio and Visual Features,* Google Inc. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5947660

[2] W.H. Tsai, H.M. Wang, "Towards Automatic Identification of Singing Language in Popular Music Recordings," Institute of Information Science, Academia Sinica, https://pdfs.semanticscholar.org/1d96/679d345c726a19125243f8c88ef1ce1b27f6.pdf

[3] M. Mehrabani and J. H. L. Hansen, "Language identification for singing," *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, 2011, pp. 4408-4411. doi: 10.1109/ICASSP.2011.5947331

[4] J. Schwenninger, R. Brueckner, D. Willett, M. Hennecke, "Language Identification in Vocal Music" https://pdfs.semanticscholar.org/e2c8/48645da41749ed8f74a674ea03e212fef8ce.pdf

[5] R. Hennequin, A. Khlif, F. Voituret, M. Moussallam: "Spleeter: A Fast And State-of-the Art

Music Source Separation Tool With Pre-trained Models" *ISMIR*, 2019.

[6] Librosa Development Team: "Vocal Separation," *Librosa Gallery on GitHub*, 2016-2017.

[7] S. Walt, S. Chris Colbert and G. Varoquaux, "The NumPy Array" *A Structure for Efficient Numerical Computation, Computing in Science & Engineering*, 13, 22-30, 2011.

[8] E. Jones, T. Oliphant, P. Peterson, et al., "SciPy: Open source scientific tools for Python", 2001.

[9] F. Pedregosa, G. Varoquaux, A. Gramfort, et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning*, vol. 12, Pp. 2825-2830, 2011.

[10] "Hearing and Testing," *Let's Talk About*, Intermountain Primary Children's Hospital, https://intermountainhealthcare.org/ext/Dcmnt?ncid=520408218

[11] K.S. Rao, S. Maity, V.R. Reddy, "Pitch Synchronous and Glottal Closure based Speech Analysis for Language Recognition," International Journal of Speech Technology, Vol. 16, Iss. 4, Pp 413-430, April 2013. https://link.springer.com/article/10.1007/s10772-013-9193-5

[12] P. Matějka, L. Burget, P. Schwarz, and J. Černocký, "Brno University of Technology system for NIST 2005 language recognition evaluation," in *Proc. IEEE Odyssey: The Speaker and Language Recognition Workshop*, 2006. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4013510&tag=1

[13] A.G. Adami, H. Hermansky, "Segmentation of Speech for Speaker and Language Recognition," OGI School of Science and Engineering, Oregon Health and Science University, Sept. 2003 https://www.isca-speech.org/archive/eurospeech_2003/e03_0841.html

[14] L. Mary, B. Yegnanarayana, "Extraction and Representation of Prosodic Features for Language and Speaker Recognition," Speech Communication, Vol. 50 , Iss. 10, Pp 782-796, May 2008. https://www.sciencedirect.com/science/article/abs/pii/S0167639308000587

[15] A.F. Martin, M.A. Pryzbocki, "NIST 2003 Langauge Recognition Evaluation," National Institute of Standards and Technology, Sept. 2003 https://www.isca-speech.org/archive/eurospeech_2003/e03_1341.html

[16] N. Dehak, P.A. Torres-Carrasquillo, D. Reynolds, R. Dehak, "Language Recognition via Ivectors and Dimensionality Reduction," MIT-CSAIL, Spoken Language System Group, Massachusits, Aug. 2011 https://www.isca-speech.org/archive/interspeech_2011/i11_0857.html

[17] L. Burget, P. Matějka, J. Cernocky, "Discriminative Training Techniques for Acoustic Language Identification," in IEEE International Conference on Acoustics, Speech, and Signal Processing, 2006. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1659994

[18] Pavel Matějka, Petr Schwarz, Jan Černocký, Pavel Chytil, "Phonotactic Language Identification using High Quality Phoneme Recognition" in Proc. Eurospeech, Sept. 2005. http://www.fit.vutbr.cz/~matejkap/publi/2005/eurospeech2005.pdf

[19] J. Navrátil, "Spoken Language Recognition - A Step Toward Multilinguality in Speech Processing," IEEE, Sept. 2001. https://ieeexplore.ieee.org/abstract/document/943345