Moodify - Detecting the Mood of Music

Emir Kaan Kırmacı [1] **Tuna Karacan** [1] **Cihad Özcan** [1]

Abstract

In this paper we propose Moodify, a mood-based music classifier that predicts the dominantly perceived emotion of the given music. Music Mood Classification is a sub-category of Music Information Retrieval that hasn't been explored much in the current literature. Researches made before usually focused on a single type of input, either audio or text. Therefore we tried to improve the performance by combining those input types. For all models, we run our experiments on a MIREX dataset. We implemented an audio-only model using deep learning techniques as well as a lyrics-only model using traditional machine learning algorithms. Lastly, considering results of preceding models, we combined both audio and lyrics and created a hybrid model using early fusion.

1. Introduction

In our daily lives we usually listen to songs that reflect our current mood and there is nothing quite as unsatisfying as listening to calming songs while we are feeling energetic and happy. That is the reason why common music listening platforms such as iTunes, Spotify, Shazam and Deezer start to take mood of songs into account when recommending songs to its users (1).

In our project we tried to make novel contrubutions to this relatively unexplored area that is mood detection. We decided to use 3 different models that use audio input, text input and both. Audio input was processed with a CNN model and text input was processed with traditional ML techniques K-Nearest Neighbors (KNN), Support Vector Machines (SVM) and Naïve Bayes Classifier (NBC). After getting the results from those 2 models, a hybrid model using both audio and text input was implemented using Early Fusion, which combines and uses the results from the older models as an input for the next layer.

2. Related Work

Music Emotion Recognition (MER) has been attracting increasing interest from the research community. Checking related works, we saw that both machine and deep learning-based methods have been used. The earlier works have focused on machine learning methods considering only verbal information like lyrics, using methods that work well with text classification such as NBC and SVM. In recent years, researchers have achieved better results by applying deep learning and working on acoustical attributes as well.

Among researches working on lyrics with traditional machine learning methods, especially one work (2) attracts our attention which uses same classification model with us and implements Naïve Bayes and SVM based classifiers with accuracies between 50.6% - 57.4% for different models. A later study (3) focuses on binary classification (happy or sad) using an NBC and reaches 88.9% accuracy on validation set. These results imply that lyrics provide a good distinction when classification is made according to quite separate moods (e.g. happy & sad) but it gets harder to classify more complex moods involving overlapping feelings. Since emotions in real life are usually very complex and it is not enough to make a binary classification, we can conclude that classical text-based classification methods are not appropriate to be applied on their own.

In comparison to that, recent deep learning-based researches have mostly considered raw audio data using neural networks like Convolutional Neural Networks(CNNs) (4), Recurrent Neural Networks(RNNs) (5) or a combination of them as CRNNs (6; 7) with Long-Short Term Memory architecture. There are also some researches using hybrid models combining lyrics and audio information based on CNNs (8) as well as SVMs (9). Moreover, some studies uses feature engineering methodologies proposing that classification can be improved by providing handcrafted features instead of raw audio data (10).

There is a well-known contest for the annual evaluation of MIR algorithms, called Music Information Retrieval Evaluation eXchange (MIREX) (11). Its results are especially meaningful for our work because it uses the same mood classification model with us. Although increasingly successful results have been obtained since 2007, state-of-the-art solutions in MIREX's mood classification from audio task are still unable to solve problem accurately, as the best algorithm has achieved only 69.8% yet. It is worth noting that one of the contestants at 2018 used same dataset with us for training and they won with 61.2% accuracy.

3. Methodology

3.1. Perspective on Mood Classification

Describing concept of emotion is not straightforward. Emotions are subjective experiences and it depends on many factors including culture, education, personal experience. There are researches showing that mood perception differ between respondents of distant cultural backgrounds(12). Although it is difficult to classify mood considering music because of the subjectivity, some researches(13) have shown that musical sounds with certain structures usually involve an acceptable degree of common emotional expression.

Moreover, there is a wide variety of methodological choices to make during classification of emotions in music. These choices result in completely different evaluation metrics, which makes the different algorithms nearly impossible to compare. Figure 1 shows various data annotation and representation choices we made in our work, as a flowchart.

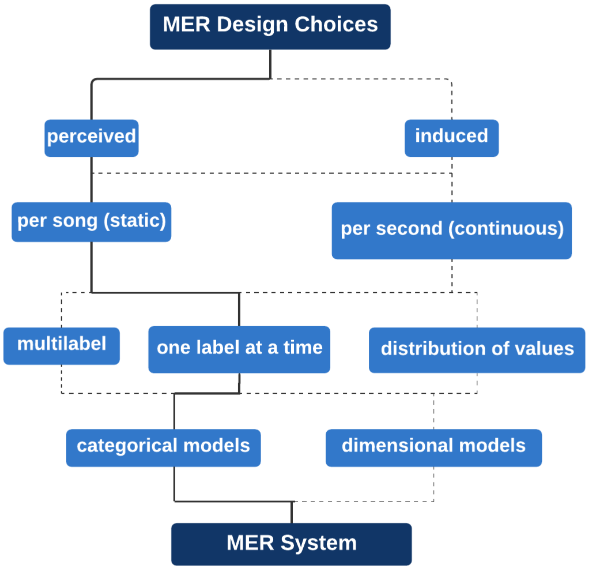


Figure 1. Methodological Choices Flowchart

Firstly, induced emotion (also known as felt emotion) is the emotion experienced by the listener whereas the perceived emotion (also known as expressed emotion) is the emotion recognized in the music.(14) Then, we prefer to classify each song with only one label according to a categorical model. Categorical models involve several distinct emotion labels, making classification simpler. On the other hand, dimensional models classifies emotions along several and independent axes in space. For example, in Russell's valence/arousal model (15) which is extensively used, valence stands for the polarity of emotion (in terms of negative and positive states) while arousal represents intensity as shown in figure 2. We stick with a categorical model to evaluate our accuracy more precisely.

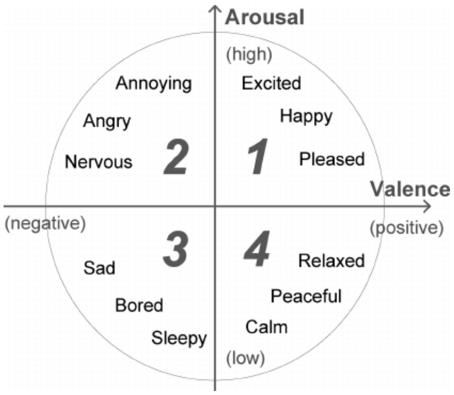


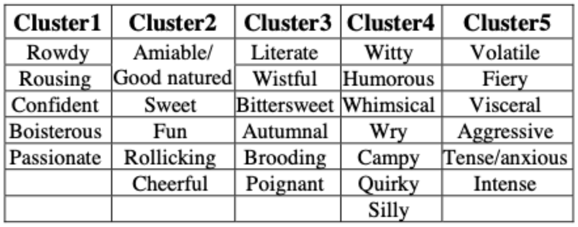
Figure 2. Russell's Valence-Arousal Model

3.2. Dataset

The dataset used in this paper is the Multi-modal MIREX-like emotion dataset (16), which consists of 903 audio clips. It also contains lyrics for 764 clips out of 903. It was built on AllMusic database and this is advantageous because annotations on AllMusic are performed by professionals while ordinary music listeners perform annotations on most of other databases.

This dataset has also been used by several competitors in Audio Music Mood Classification task of MIREX. So, we can compare our results to those implementations in order to evaluate whether our model's accuracy is successful enough.

The dataset is composed of 5 main clusters, each of which includes some adjectives. As being in the same cluster, those adjectives are similar to each other, and describe same emotional state. As an example, the adjectives Silly, Whimsical and Humorous belong to the same cluster and correspond to the same class during classification. Figure 3 shows all the adjectives and the clusters in which they belong.



Cluster1	Cluster2	Cluster3	Cluster4	Cluster5
Rowdy	Amiable/	Literate	Witty	Volatile
Rousing	Good natured	Wistful	Humorous	Fiery
Confident	Sweet	Bittersweet	Whimsical	Visceral
Boisterous	Fun	Autumnal	Wry	Aggressive
Passionate	Rollicking	Brooding	Campy	Tense/anxious
	Cheerful	Poignant	Quirky	Intense
			Silly	

Figure 3. Emotion Clusters and Their Constituents

Since using adjectives would probably overfit on a small dataset such as ours, we only used the clusters as labels.

3.3. The Models

We implemented 3 models which are the Audio Model, Lyrics Model and the Hybrid Model. The Audio Model and the Lyrics Model use audio and lyrical data respectively, while the Hybrid Model uses both. The baselines for the Audio Model and Lyrics Model are built on top of and Panda et al(16) and Wenhao Bian's(17) work. We used Convolutional Neural Networks (CNNs) for the Audio Model and 3 machine learning methods for Lyrics Model which are K-Nearest Neighbors, Naïve Bayes Classifier and Support Vector Machines. We implemented the Hybrid Model from scratch, by combining CNN architecture (which processes audio data) with MLP architecture (which processes lyrics). We thought that using both textual data and audio data together in the Hybrid Model could improve the overall accuracy.

3.3.1. The Audio Model

The Audio Model, as its name suggests, only takes the audio data as input which consists of 30 seconds long 903 tracks in total. We used CNNs for our audio-only model, as they're one of the more common and efficient models in Music Information Retrieval tasks because of their ability to match patterns in audio data effectively. Previous research about music mood classification also deployed CNN methods, and CNNs were used by the competitors in MIREX as well. The closest work that we can compare our model was implemented in MIREX 2018, where one of the contestants won using a CNN with an accuracy of 61.2% (17). Although accuracy looks low on paper, as said before, getting high accuracies while classifying complex moods is a rather tough task, as state-of-the-art solutions in MIREX has the accuracy of 69.8% at most. Considering it as a baseline, we used a finely-tuned version of the mentioned model from 2018.

As inputs to our CNN model, we used Mel Spectrograms of the audio data. A spectrogram is the visual representation of the frequency spectrums of an audio with respect to time. Getting the spectrogram of an audio is done by doing a Fast Fourier Transform (FFT), which brings the audio from time domain (amplitude wrt. time) to frequency domain (frequencies wrt. time). Mel spectrograms are spectrograms that are converted to mel scale, in order to better perceive the hearing range of humans. Hence, mel spectrograms are a widely used input type in Music Information Retrieval tasks. They are also used as input in mentioned baseline implementation from MIREX 2018.

3.3.2. The Lyrics Model

In the Lyrics Model, we implemented multiple machine learning algorithms to test their accuracies on the lyrical data, similar to the works of Panda et al.(16) and Thanh & Shirai(2). Those algorithms are K-Nearest Neighbors (KNN), Naïve Bayes Classifier and Support Vector Machines (SVM). All of those algorithms are commonly used in Natural Language Processing (NLP) applications such as sentiment analysis. Since the Lyrics Model is more closely related with analyzing and classifying the textual data, we thought that using algorithms that are effective in NLP-based tasks could give us better accuracies.

We tested each of those 3 algorithms separately on the lyrics dataset which has lyrics for 764 tracks, and compared their accuracies.

3.3.3. The Hybrid Model

For the last part of the project, we used the Hybrid Model which combines the outputs of both lyrical and audio data within feature-level (Early Fusion). From our research, we didn't see a hybrid architecture similar to ours in MIREX and the only research that was similar to our work was of Delbouys et al. (8), where they used a bimodal arechitecture using CNNs for audio data and LSTMs for lyrical data. Still, their classification method for emotions was quite different than us. They use a dimensional model which evaluates emotions with continuous values along two dimensions, instead of our categorical model which uses discrete values (classes). This does make our research more distinct compared to previous work.

CNN architecture is used for the audio-part and MLP architecture is used for the lyrics part. After getting their respective results, they are combined and given as an input to another MLP model.

4. Experiments

4.1. Evaluation Criterias

As explained in methodology section, we defined the problem as a multi-class classification problem considering 5 classes which are clusters consisting of similar adjectives for moods. We used categorical cross-entropy loss for our model's loss function. Additionally, results were evaluated by comparing our model's accuracy to other studies using same dataset or at least same approaches, in order to have a better understanding about its success.

The dataset was divided into training and test parts for both of the models, with 90% of the data being the training data and the remaining 10% being the test data. For the Lyrics Model, 10-fold cross-validation was used for hyperparameter optimization. For all the models, confusion matrices are used with in order to specify which classes were predicted better.

4.2. The Audio Model

The first model we implemented in our project was the Audio Model. We read the 903 tracks divided by their clusters and their classes, converted the data into mel spectrograms and saved them into JSON files as Pandas DataFrames, in order to simplify the reading process. Each input in the DataFrame has cluster, class and mel spectrogram data. The mel spectrograms have 96 mel-bands and are pre-processed so that all of them have the same frame lengths. The frames have been zero-padded to the spectrogram with the longest frame length.

After saving the data as JSON files, we read and merged them into one dataset. The class data is dropped since using it would most likely cause overfitting, and the cluster data is transformed from text data to numerical data in order to be used by the model. After that, for the initial tests the dataset is splitted into training, validation and test data (80%, 10% and 10%) instead of 90% training and 10% test data.

After the preparation of the dataset, the CNN model is implemented with 5 convolutional layers. The filter counts for the layers are 32, 128, 128, 192, 256 respectively, with each filter having the size 3x3. In each layer, Rectified Linear Unit (ReLU) is also used as the activation function. The convolutional layers are zero-padded in order to preserve the dimension counts. Every convolutional layer is followed by a Batch Normalization layer and a Max. Pooling layer. After the convolutional layers, a Dropout layer with 0.5 rate is used to prevent overfitting. Finally, the data is flattened and sent to the output layer with size 5 (cluster size) and the Softmax activation and Cross-Entropy loss functions are used.

After setting up the model structure and training the model with 300 epochs with RMSProp as the optimizer, we tested the model on the test data. Unfortunately, the accuracy was lower than what we expected with a value of 31%, compared to the original value of the CNN model used in MIREX 2018 which is 61.2%.

It seems like the model is overfitting to the training data a lot. This might be because of the training data being too small, or we didn't implement the model correctly. We think of the former, as we're simply using 20% of the data for test and validation although dataset involves only 903 samples. But since the original result for a model like this is at 61.2%, there is a high chance that the model isn't implemented correctly as well.

We also tested the model without Dropout and Batch Normalization layers to see their effects on the training. You can see the accuracies in the table below. Do note that using Batch Normalization stabilized errors at each epoch a lot.

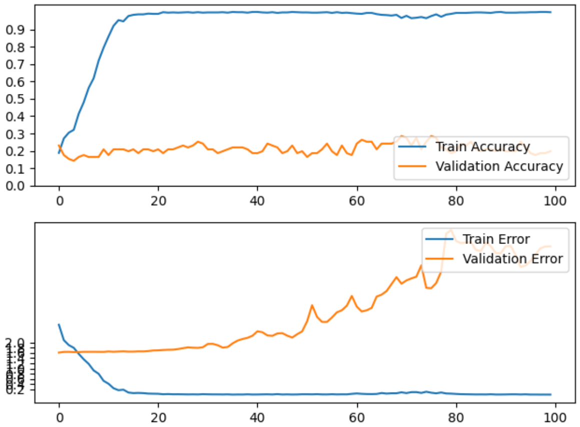


Figure 4. Accuracy/Error for the Audio Model

Results	
Model Used	**Accuracy**(%)
Model	31
Model w/o Dropout	28
Model w/o Batch Normalization	28

4.3. Improving the Audio Model

To improve the audio model, we did some changes to both the architecture of our model and the data itself. Our main goal here was to reduce overfitting since it was the main problem of the model (model learns the training data too fast and doesn't really learn the testing data).

In our preliminary tests, the input shape of the CNN was (903, 1, 96, (max. frame length)). From the perspective of a CNN, this can be seen as a 1x96 image with a channel size of (max. frame length). Since CNNs are mostly used with small channel sizes (1 for greyscale pixels and 3 for RGB pixels), we reshaped the data as (903, 96, (max. frame length), 1). This is more similar to the image inputs given to a CNN.

In addition to the shape change, instead of using only one dropout layer at the end of the convolutional layers, we added dropout layers after each convolutional layer with increasing dropout rates. Since the first layers of the model has less filters and dropping some of the inputs in the first layers would also affect the other layers quite a bit, starting with a lower dropout rate and increasing it in the later layers was our preferred choice in our new architecture.

We also decreased the learning rate of model from 0.001 to 0.0001 since training accuracy was getting around 98% accuracy really early (around 10 epochs). A high learning rate could also mean that model misses the global optima while applying the backpropagation algorithm.

And finally, we changed the train-test split of the data to 90%-10%. Since our dataset is quite small, we thought that adding more samples to the training split would be more efficient to train the model.

After the improvements to our model, the accuracy improved from 0.31 up to 0.46. The final architecture of the model can be seen in Figure 5.

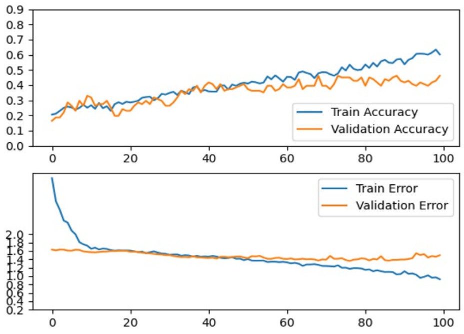


Figure 5. Accuracy/Error for the updated Audio Model

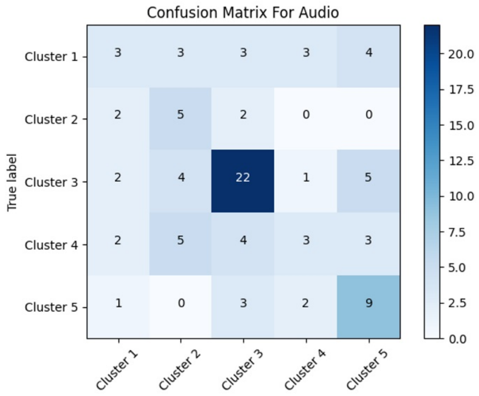


Figure 6. Confusion matrix for the updated Audio Model

Best Accuracy for Audio Model: 0.46

We can see that even though the generalization of our model is still isn't that high, those improvements did reduce the overfitting as we can see from the acccuracy/error plots above. Our results could be improved even further with fine-tuning the model more and applying preprocessing methods such as data augmentation or splitting the data into smaller parts in order to create more samples. Changing the optimizer could also work, but our tests with Adam optimizer didn't give us better results than RMSProp. But overall, we think that with a limited dataset such as Multi-modal MIREX-like emotion dataset (16) and the difficulty of music mood classification for computers as a whole, the results we got from our model isn't exactly that bad.

4.4. The Lyrics Model

For lyrics, we implemented K-Nearest Neighbors (KNN), Naïve Bayes Classifier, Support Vector Machines (SVM) algorithms and compared results to find the best model. Besides showing machine learning algorithms' accuracy on lyrics, other main objective of constructing Lyrics Models is finding out the most appropriate feature extraction methods for our lyrics dataset. Because same feature extraction process is employed for Hybrid Model, too. So that, we tried different approaches for data extraction and evaluated their accuracy implementing grid search with 10-fold cross-validation on various model parameters.

Initially, we read lyrics of 764 tracks as strings and stored corresponding labels. Then all lyrics were pre-processed by; converting to lowercase letters, removing punctuations and numbers, removing stopwords considering those involved by NLTK corpus.

Lyrics features were extracted using Bag of Words (BoW) model and vectorized counts of n-grams in whole dataset. For further data processing before vectorization, we considered different approaches as parameters and tried all of them utilizing grid search. These parameters are provided below:

- Tf-idf transformation: [True, False]

- N-gram range: [(1, 1), (1, 2), (2, 2), (1, 3), (2, 3), (3, 3)]

- Text normalization method: [stemming, lemmatization, None]

Along with data processing methods, we optimized each models' specific parameters. All tried parameters are given below:

For Support Vector Machine:

- Regularization parameter(a.k.a. 'c' or penalty): [0.1, 1, 10, 100, 1000]

- Kernel coefficient(gamma): [1, 0.1, 0.01, 'scale']

- Kernel: [Gaussian radial basis, polynomial with degree 3, linear]

For Multinomial Naive Bayes:

- Laplace smoothing parameter: [1, 0.1, 0.01, 0.001, 0.0001]

For K-Nearest Neighbors:

- Number of Neighbors: [3, 5, 8, 12, 15, 20, 29, 40]

- Weight function: [uniform, distance]

- Distance Metric: [euclidean, manhattan]

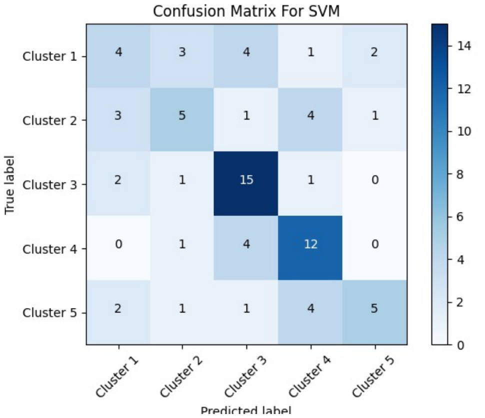
Moreover, we considered train-test split ratio of dataset as another hyperparameter and repeated all steps for different test sizes (10%, 15% and 20%). Since dataset size is limited, this repetition with different test sizes also minimized chance factor on parameters' effect on accuracy.

According to the results of grid search, best accuracies are obtained when tf-idf is used on unigrams(1,1) and stemming is applied for text normalization. Also, we saw that the best train-test split ratio is (90%-10%).

The best accuracies, optimized parameters and confusion matrices for each algorithm are given below:

Best SVM Classifier:

- Accuracy: 53.2%
- Regularization parameter: 1
- Kernel: linear function

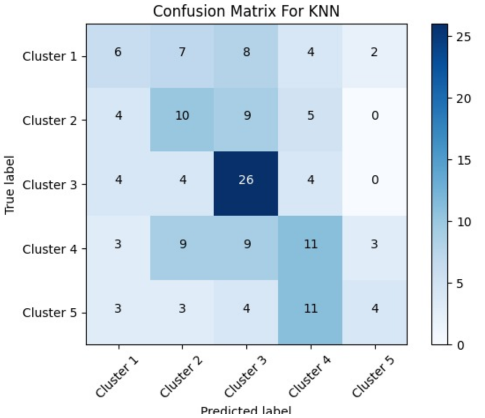


	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Cluster 1	4	3	4	1	2
Cluster 2	3	5	1	4	1
Cluster 3	2	1	15	1	0
Cluster 4	0	1	4	12	0
Cluster 5	2	1	1	4	5

Figure 7. Confusion Matrix of Best SVM Classifier

Best KNN Classifier:

- Accuracy: 37.3%
- Number of Neighbors: 29
- Weight function: distance
- Distance Metric: euclidean

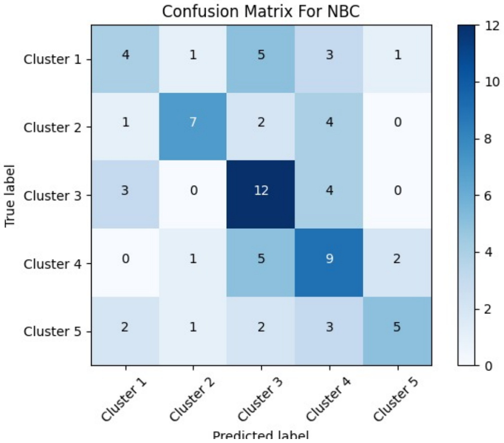


	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Cluster 1	6	7	8	4	2
Cluster 2	4	10	9	5	0
Cluster 3	4	4	26	4	0
Cluster 4	3	9	9	11	3
Cluster 5	3	3	4	11	4

Figure 8. Confusion Matrix of Best KNN Classifier

Best Naive Bayes Classifier:

- Accuracy: 48.1%
- Laplace smoothing parameter: 0.01



	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Cluster 1	4	1	5	3	1
Cluster 2	1	7	2	4	0
Cluster 3	3	0	12	4	0
Cluster 4	0	1	5	9	2
Cluster 5	2	1	2	3	5

Figure 9. Confusion Matrix of Best NBC Classifier

KNN algorithm turns out to be ineffective compared to others. While Naive Bayes Classifier's accuracy (48.1%) is not bad, SVM-based classifier is the most successful one with 53.2% accuracy. These results don't seem that accurate but we can say that they are acceptable for our dataset. Baseline research(2) has accuracies between 50.6% and 57.4% obtained by Naive Bayes and SVM-based classifiers with different models, although they do feature engineering and utilizes metadata like artist and title information as well as weighting some parts of lyrics (e.g. chorus part). So that, we can improve our model's accuracies if we use metadata similar to those used in baseline research.

4.5. The Hybrid Model

From our results, we saw that the lyrics model performed better than the audio model. As we also have audio data in our dataset, it would be nice if we could have a way to use both audio and lyrics together in a model in order to get better accuracies. So, we designed a hybrid model which takes both the audio and lyrics as inputs.

The hybrid model itself consists of 3 parts. The first part is a CNN model which takes the audio data as input, the second part is a Multi-Layer Perceptron (MLP) model which takes the lyrical data as input, and the third part is a hybrid model which takes the output of both the audio and lyrical models, combines them and gives them as an input to its first layer. Since the combination of the outputs of the 2 different models is done at the feature-level (outputs of each model are given as an input to the hybrid model), our hybrid model is using Early Fusion.

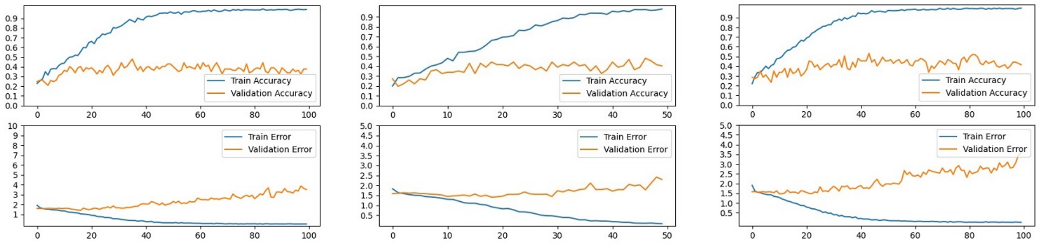


Figure 10. Accuracy/Error for the Hybrid Model with a balanced/audio-heavy/lyrics-heavy architecture.

For the audio-part of the model, a CNN architecture similar to the Audio Model is used. There are 4 convolutional layers with 3x3 filters with 32, 128, 128, 192 filter counts and ReLU activation function. Each convolutional layer is also followed by a Batch Normalization layer, a Max. Pooling layer and a Dropout layer. Dropout rates are set in an increasing fashion (from 0.034 to 0.25) because of the same reasons given in Improving the Audio Model section. After the convolutional layers, the data is flattened as the last layer. Since the results we got from the Lyrics Model was better than the Audio Model, we thought that making the audio-part of the hybrid model simpler while making the lyrics-part more complex would give us better results (more on the results later).

For the lyrics-part of the model, a simple MLP model is used with 2 dense layers with 64 and 128 neurons. A dropout layer with a rate of 0.25 is also used.

Finally, to create the hybrid model, the outputs of the audio-part and lyrics-part are combined using Tensorflow's concatenate layer, and the resulting output is given as an input to the Hybrid Model's first layer. The hybrid model consists of 1 dense layer with 128 neurons and 1 output layer with 5 neurons (cluster size). Softmax activation function and Cross-Entropy loss functions are used for the output layer, similar to the Audio Model before.

The results for the hybrid model can be seen in Figure 10.

Best Accuracy for the Hybrid Model: 0.42

4.6. Effects of Audio and Lyrics on the Hybrid Model

We tried 3 different approaches with the hybrid model in order to see the effects of the audio and lyrical data.

Our first approach was to use the Audio Model itself for the audio-part of the hybrid model, which means that it had 5 convolutional layers and a dense layer at the end. The lyrics-part was the same as we said before (2 dense layers with 1 dropout layer). As our first result had an accuracy of 0.38 with max. accuracy as 0.48, we tried to change the architecture of the model in order to get better results.

First, since the Lyrics Model performed better than the Audio Model, we tried to simplify the audio-part's architecture by dropping the last convolutional and dense layer from the CNN. By doing this, we aimed at prioritizing lyrical data before the audio data since lyrics seemed to be better at predicting compared to the audio. This change improved our results up to 0.42 and the max. accuracy became 0.53.

We also tried simplifying lyrics-part instead of audio-part by dropping the last dense layer and the dropout layer from the MLP architecture. The accuracy after this change was 0.40, with max. accuracy being 0.44.

From our results, we can see that the model that has a simpler CNN architecture gave us better results, but the overall results of all the models are close to each other. One main thing that all the models suffer from is the overfitting which is way more than the Audio Model has had. We think that the reasons for this high amount of overfitting can be tied to some of the constraints we had. First, since we're using both audio and lyrics, we needed to give the audio of a song and its corresponding lyrics together. And as not all audio files had lyrics (764 lyrics files compared to 903), we had to drop the audio files that did not have lyrics, therefore our dataset was smaller. In addition to that, since we used 2 different models for audio-part and lyrics-part of the hybrid model our model could've been too complex since we don't have much training data. But in our tests, using simpler models for audio and lyrics-part together gave us worse results.

5. Conclusion

From our results, we saw that the Lyrics Model gave us the best results at 0.53. Since mood in audio is rather subjective and more abstract than mood in lyrics, we can see why it gave us better results (as tied with a certain emotion is more easier to classify). Despite being the best result of the model, it could still be improved by using more up-to-date methods such as LSTM's or even Transformer Models like BERT (Bidirectional Encoder Representations from Transformers).

Even though lyrics was our best result, the Audio Model's best result is not that far off at 0.46. We do think that the results for the audio could be improved quite a bit. First of all, since our data is limited and prone to overfitting (though this isn't as high for the Audio Model compared to the Hybrid Model), we could use pre-processing techniques such as data augmentation to be able to create noisy data, so that the model doesn't overfit to the training data. We could also split the audio tracks into smaller ones in order to technically increase the dataset (though this could come with its fair share of issues). For the model itself, we could fine-tune the layer counts, filter sizes and such in order to improve the accuracy as although our model is kind of fine-tuned, there's still quite a bit of room for improvement. In addition to those, there's also the fact that we use Mel Spectrograms as the audio input. Since we use them similar to how images are used to train CNNs, maybe using a different type of audio input (such as MFCCs) could work better since using audio in an image-like fashion doesn't seem to work too well for us.

Our worst result was from Hybrid Model with 0.42 accuracy. Although combining both model and lyrics is a good idea on paper, we could argue that combining 2 different models together into the start of another model could be too complex for our rather small dataset. So with a bigger dataset and a more finely-tuned model with maybe a different architecture (such as LSTMs for lyrics), we could get better accuracies for our Hybrid Model.

We do think that improving the results of the Hybrid Model is quite an interesting subject, and we can follow different paths in order to accomplish it.

References

[1] Murray Stassen. Spotify Is Testing A Feature That Recommends Music To Match The 'Moment And Mood' Of A Photo, 03 2020.

[2] Trung-Thanh Dang and Kiyoaki Shirai. Machine Learning Approaches for Mood Classification of Songs toward Music Search Engine. *Knowledge and Systems Engineering, International Conference on*, 0:144–149, 10 2009.

[3] Sebastian Raschka. MusicMood: Predicting the mood of music from song lyrics using machine learning. 11 2016.

[4] Rajib Sarkar et al. Recognition of emotion in music based on deep convolutional neural network. *Multimedia Tools and Applications*, 79:765–783, 2019.

[5] Huaping Liu, Yong Fang, and Qinghua Huang. Music Emotion Recognition Using a Variant of Recurrent Neural Network. 2019.

[6] Zijing Gao et al. A Novel Music Emotion Recognition Model for Scratch-generated Music. In *2020 International Wireless Communications and Mobile Computing (IWCMC)*, pages 1794–1799, 2020.

[7] Miroslav Malík et al. Stacked Convolutional and Recurrent Neural Networks for Music Emotion Recognition. 2017.

[8] Rémi Delbouys et al. Music Mood Detection Based On Audio And Lyrics With Deep Neural Net. *CoRR*, abs/1809.07276, 2018.

[9] Xiao Hu, Kahyun Choi, and J. Downie. A framework for evaluating multimodal music mood classification. *Journal of the Association for Information Science and Technology*, 68, 2016.

[10] Renato Panda, Ricardo Malheiro, and Rui Pedro Paiva. Audio Features for Music Emotion Recognition: a Survey. *IEEE Transactions on Affective Computing*, PP:1–1, 10 2020.

[11] Mirex homepage. `https://www.music-ir.org/mirex/wiki/MIREX_HOME`, Last accessed on 2021-05-09.

[12] Xiao Hu and J.H. Lee. A Cross-cultural study of music mood perception between American and Chinese listeners. *Proceedings of the 13th International Society for Music Information Retrieval Conference, ISMIR 2012*, pages 535–540, 2012.

[13] Carol Krumhansl. Music: A Link Between Cognition and Emotion. volume 11, 2002.

[14] Alf Gabrielsson. Emotion Perceived and Emotion Felt: Same or Different? *Musicae Scientiae*, 5:123–147, 01 2002.

[15] James Russell. A Circumplex Model of Affect. *Journal of Personality and Social Psychology*, 39:1161–1178, 1980.

[16] Renato Panda, Ricardo Malheiro, Bruno Rocha, António Oliveira, and Rui Pedro Paiva. Multi-Modal Music Emotion Recognition: A New Dataset, Methodology and Comparative Analysis. 2013.

[17] 2018 mirex results on audio mood classification task. `https://www.music-ir.org/nema_out/mirex2018/results/act/mood_report/summary.html`, Last accessed on 2021-05-10.