PREDICTING EUROVISION SCORES BASED ON LYRICS AND AUDIO FEATURES

RAHMI AYDIN

THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN DATA SCIENCE & SOCIETY
AT THE SCHOOL OF HUMANITIES AND DIGITAL SCIENCES
OF TILBURG UNIVERSITY

STUDENT NUMBER

665759

COMMITTEE

dr. Raquel G. Alhama
dr. Afra Alishahi

LOCATION

Tilburg University
School of Humanities and Digital Sciences
Department of Cognitive Science &
Artificial Intelligence
Tilburg, The Netherlands

DATE

January 13, 2022

WORD COUNT

7878

ACKNOWLEDGMENTS

I want to thank my supervisor Raquel G. Alhama for her guidance during the whole process. Also, I would like to thank my family, especially my caring mother and father, for all their emotional support. Finally, thanks to my friends Sena, Mahir, and Hakan for listening to all my whinings.

PREDICTING EUROVISION SCORES BASED ON LYRICS AND AUDIO FEATURES

RAHMI AYDIN

Abstract

Eurovision Song Contest is one of the biggest musical organizations in Europe. Every year, millions of people eagerly await which song will be the winner. Although multiple studies examine hit songs and their characteristics, Eurovision is a relatively new area for research. This study aims to predict the score of a song in the contest based on its lyrics and audio features. For this purpose, random forest and neural network models are trained with BERT embeddings of translated (English) lyrics and audio descriptors retrieved from Spotify. In addition, a joint neural network model is fed with both lyrics and audio. All models have better performance compared to baseline. The most successful model is random forest trained with audio features. However, models' ability to predict is found to be minimal. The study also suggests that loudness is the most important feature in the random forest model and combining two neural networks as a joint model boosts the model's predictive power.

1 INTRODUCTION

What is behind the success of a song at the Eurovision Song Contest? How come some songs don't get any points, while others get everyone's likes? Is there a perfect recipe for a winner song? This thesis aims to predict the success of Eurovision songs based on their lyrics and audio features and find out which properties of a song lead to success in this competition and to what extent.

The Eurovision Song Contest (ESC), widely known simply as Eurovision, is an international competition organized annually with participants representing mainly European countries. Eurovision has been one of the most popular organizations in Europe for years. The journey of the competition began in 1956 with the participation of seven countries. The popularity of the organization has grown immensely throughout the years.

Thirty-nine countries participated in the competition held in 2021, and 183 million watched the show (Grabor, 2021).

Every year, fans and music authorities all over the world try to find out song from which country will be the winner. Bets are opened months in advance. Contestants are striving to produce and present the best piece in order to win. Of course, it can't be claimed that only the song matters for a country to collect the points since other factors such as politics and relationships also apply. However, unlike the other factors, the properties of a song could be controlled to generate the potential winner. Eurovision has distinctive aspects compared to the other international and national ranking charts such as The Billboard Hot 100. For instance, most Eurovision songs carry the cultural characteristics of the performing country, including traditional sounds. Furthermore, the target audience is predominantly European, and the ranking does not solely depend on listeners. Although the scoring system of Eurovision has been changed several times with varying rates of televoting and professional juries' points, it can be claimed that there is a pattern behind the ranking. Revealing the success-driven features of the songs can also guide the music industry by shedding light on the listeners' and juries' preferences.

Predicting the Eurovision results is a problem getting growing attention as more data and different techniques become available. For example, in a paper by Demergis (2019), tweets posted during the 2019 Eurovision were analyzed to predict the results. He predicted the rank for each contestant and resulted in a significant correlation between the sentiment of tweets and televoting. In addition, there are a few more research used tweet analysis for the prediction problem Kakouris, Theocharis, Vlastos, and Memon (2016); Kumpulainen et al. (2020). However, the prediction of results by using lyrics and audio properties of the song has not been researched yet. Nonetheless, various studies have used similar properties to predict the hit songs. For example, Kim and Oh (2021) investigated which acoustic properties of a song could improve the model's capability to predict the hit songs in Billboard Lists. They used the acoustic features extracted by Spotify Web API and concluded that granular data offered by music analysis tools have a significant predictive value. Other papers also tried to predict the hit songs with various techniques and datasets (Middlebrook & Sheik, 2019; Yu, Yang, Hung, & Chen, 2017). The literature regarding the subject has been examined extensively in the following section.

For the purpose of this thesis main research question has been constructed as such :

> *Can we predict the score of a song at the Eurovision Song Contest based on its lyrics and audio features?*

In order to answer the research question, three sub-questions will be examined :

RQ1 *Can we predict the score based on audio features, if so, to what extent?*

RQ2 *Can we predict the score based on lyrics, if so, to what extent?*

RQ3 *Can we predict the score based on the combination of lyrics and audio features, if so, to what extent?*

The study aims to discover which properties are the most valuable in the prediction problem by answering these questions.For instance, whether lyrics or audio features have more predictive power? Or combined model works better? More specifically, does tempo or acousticness have a more significant effect?

Random forest and neural network models have been created in order to address the research questions and the following sub-questions. When compared to the baseline, all models performed better. Random forest trained with audio characteristics was the most successful model. The capacity of models to predict, on the other hand, is shown to be limited. The study also revealed that loudness is the most relevant attribute in the random forest model, and merging two neural networks as a joint model improves the model's predictive capacity.

2 RELATED WORK

One point is worth mentioning before starting the literature review. We couldn't find any scientific research that in-depth analyze the lyric and/or audio features of Eurovision songs to predict success. For this reason, we divided this part into two sections: goal and method. First of all, studies with a similar aim to our research will be examined, then studies with similar data and techniques will be focused on.

2.1 *Eurovision*

Predicting scores of songs from data in Eurovision has not been widely researched. One of the earliest studies was done by Ochoa, Hernández, Sánchez, Muñoz-Zavala, and Ponce (2008) . They tried to predict the rank of a country in the first Eurovision event that they participated in. For this task, they applied data mining techniques to analyze voting patterns in the contest. They concluded that performers' and voting nations' linguistic and cultural proximity significantly influence the success of a country. This study is significant because it is one of the first attempts to predict

a country's rank in Eurovision from data. Interestingly, other researches conducted on the topic mainly focused on Twitter data. Kakouris et al. (2016) applied opinion mining and analyzed the Twitter data to find hidden patterns and predict Eurovision 2014 winner. They showed that their findings are highly dependent on the period of the harvested data, and they are not strong enough to determine but still could help predict the winner. Stieglitz, Meske, Ross, and Mirbabaie (2020) also emphasized the importance of selected time periods in their study that was used 2015-2016 Twitter data to predict results in the contest. Since 2016, data mining techniques have improved, and social media usage has increased drastically. For the same purpose, Demergis (2019) collected English and Spanish-language tweets and analyzed their sentiment to predict the ranks of countries during televoting in 2019 Eurovision. The study demonstrated a substantial correlation between the mood retrieved from Eurovision fans' tweets and their preferences during the televoting. Compared to the approach by Demergis (2019), Kumpulainen et al. (2020) employed a simple sentiment lexicon-based strategy and sentiment analysis using BERT based deep learning method (Y. Liu et al., 2019). They came to the conclusion that tweets are strongly correlated with the voting behavior of the audience, as the deep learning method resulted in a stronger correlation.

Literature focuses mainly on external factors such as sentiment in social media when predicting the countries' scores from data. In that regard, this project aims to fill the gap by comprehensively analyzing internal factors. Thus, we intend to use the lyrics and audio features of the songs from the beginning of the competition as a new approach to predict a country's rank.

2.2 *Hit Song Science*

Although the prediction of Eurovision winners from data is a new study area, prediction of the hit songs is studied more frequently. As a research topic, Hit Song Science (HSS) aims to predict whether a song will be commercially successful before its release by analyzing the specific features (Middlebrook & Sheik, 2019). Studies mainly focus on two aspects, namely internal and external factors. Internal factors represent the lyrics and the audio features such as rhythm, tune, and tempo. On the other hand, external factors include social media reactions and the song's metadata, such as composer and release date. For instance, Zangerle, Pichl, Hupfauf, and Specht (2016) showed that Twitter posts have predictive power in identifying future hit songs. Despite the promising results of the external factors' predictive capabilities, we will focus solely on the internal factors

in our study since we try to find out lyrics' and audio features' predictive capabilities and compare them.

Most of the studies focus on internal factors with or without metadata, and the hit song prediction was usually treated as a classification and regression problem. Lee and Lee (2018) used classification methods to predict the popularity metrics of a song. In order to achieve this task, they used only acoustic data and found out that Support Vector Machine (SVM) was the most successful in predicting the popularity metrics. The result showed that although the accuracy is not very high (around %57), the model with only audio data is significantly better than a random score prediction. Araujo, de Cristo, and Giusti (2019) tried to classify the song according to whether they will climb to Spotify's Top 50 Global chart in advance. For this task, they used the acoustic properties of the song and its past information from the platform's chart. The experiment yielded promising results in predicting the song's success two months advance with the SVM classifier. Middlebrook and Sheik (2019) also used audio features to predict hit songs. They extracted audio features of 1.8 million hit and non-hit songs via Spotify Web API and trained their models. The most significant contribution of this study is to train their model with a bigger dataset compared to datasets used in prior similar studies. Although the accuracy scores of all trained models are above 80 percent, the most successful model was Random Forest. Kim and Oh (2021) also used audio descriptors of songs provided by Spotify in their studies. They examined how acoustic features improve the prediction of the top-10 popular songs and discovered that some features considerably boost the model's predictive power. This result also encourages using detailed data provided by music intelligence technology such as Spotify's.

Audio features are widely used to predict hit songs, but other internal factors are mostly disregarded. Demetriou, Jansson, Kumar, and Bittner (2018) investigated the most influencing attributes of a song when it comes to liking or disliking it. According to respondents, the study concluded that lyrics are one of the most important features of a song. This result is encouraging to also include lyrics features in predicting the hit songs. In that regard, Raza and Nanath (2020) combined sentiment analysis of lyrics along with the audio features to have a more robust model in the prediction task. However, they took a stand that the prediction of hit songs is not a data science activity yet.

Deep learning techniques were also used in the hit song prediction problem. In that regard, Yang, Chou, Liu, Yang, and Chen (2017) conducted one of the earliest studies. They used different deep learning models in audio-based hit song prediction and concluded that deep structures are better than shallow structures in predicting popularity. Yu et al. (2017)

extended the previous study by treating the hit song problem also as a ranking problem. They proposed a convolutional neural network (CNN) model that jointly learns hit songs' scores and relative ranks. Another joint model was introduced by Zangerle, Vötter, Huber, and Yang (2019). They proposed a model that jointly learns low- and high-level audio features in the regression task of predicting hit songs. They concluded that the proposed architecture outperforms the baseline model. Following a similar strategy, Sharma (2020) utilized the joint neural network model with audio features and lyrics. Audio features have been designated as low-level and lyrics as the high-level representation of a song. The collaborative model significantly outperformed the separate audio and lyrics models.

Despite the fact that the hit song prediction problem is different from predicting the score of a song in Eurovision, the properties of a song and potential techniques are the same. Based on previous studies, the audio features of a song could help determine its success. Furthermore, the predictive power of lyrics was not analyzed sufficiently in literature in contrast to encouraging results of related studies (Demetriou et al., 2018). Most of the studies solely focused on audio features. Moreover, previous studies show that machine learning models such as Random Forest and joint neural network models are successful in the hit song prediction tasks. Joint neural network models have the capability to reflect the predictive power of a song's different features proportionally. All these findings have affected our decisions when designing models.

3 METHOD

This section will introduce the methods chosen for our study. A more detailed explanation of the model setup will be discussed in the Experimental Setup section.

3.1 *BERT*

Traditional word embedding and contextual embedding techniques try to transform words into vector representations in a document. These representations of words can be used as input in machine learning tasks. Non-contextual word embedding techniques such as Glove and Word2Vec try to form a global vocabulary with the unique words in the text and create a representation based on how frequently the word appeared (Si, Wang, Xu, & Roberts, 2019). However, this approach ignores the possibility of words having different meanings in different contexts. For instance, bear has two separate meanings in the sentences "I saw a bear in the forest" and "I can't bear it" but has the exact representation in the embedding. On the

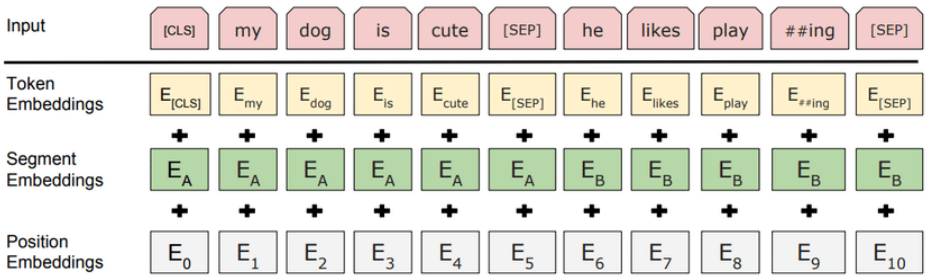

Figure 1: Input representation in BERT. The sum of the token embeddings, segmentation embeddings, and position embeddings is the input embeddings. Devlin et al. (2018)

other hand, contextual embeddings take this situation into account and represent each word depending on its context. As a result, word usage in various contexts is captured, and knowledge that transfers between languages is encoded (Q. Liu, Kusner, & Blunsom, 2020).

In this project, we use BERT embeddings in order to transform lyrics into vector representations. BERT (Bidirectional Encoder Representations from Transformers) is a contextual word embedding technique designed to help models learn the word meaning from context by jointly using the surrounding text (Devlin et al., 2018). Compared to other traditional embedding models, it showed extraordinary success in natural language processing (NLP) tasks (Tenney, Das, & Pavlick, 2019). Moreover, BERT easily handles various types of NLP tasks. To bring BERT this versatility, developers created a set of rules in pre-processing step to represent the input for the model. Input embeddings consist of 1) positional, 2) segment, and 3) token embeddings. This architecture allows storing multiple layers of information regarding input text (see Figure 1). With the given input, the framework of the model consists of two steps: pre-training and fine-tuning. In the pre-training step, developers use Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) as unsupervised tasks in contrast to other pre-training tasks such as word representations and sentence embedding methods (Peters, Ruder, & Smith, 2019). The MLM task enables BERT to train bidirectionally by predicting the masked tokens with the tokens in context. This way, the model doesn't predict the next word but predicts the random missing word from the sequence. The second task, NSP, is a binary classification task, predicting if B is the following sentence of the given sentence A. As the MLM task finds the relationship between words, the NSP task helps understand the relationship between sentences. For English, 3,300 million words were used as pre-training corpus. After these two tasks are completed, the model is ready to fine-tune. Fine-tuning

pre-trained models boosts performance on various NLP tasks, and BERT is one of the most successful models for this procedure (T. Zhang, Wu, Katiyar, Weinberger, & Artzi, 2020). In the application, Devlin et al. (2018) announced two sizes of BERT models for English: BERT-base and BERT-large. The BERT-base features 12 encoder layers, each with 12 attention heads. On the other hand, BERT-large is made up of 24 encoder layers, each of which has 16 attention heads. Their hidden sizes are 768 and 1028, respectively.

To summarize, BERT is an effective tool for word embeddings. It is suitable for a wide range of NLP tasks and is highly successful in uncovering dependencies between words and sentences. Therefore, we chose BERT to transform lyrics to word embeddings while training our models for the prediction task. In section 4 , we will give greater detail on the feature choices while applying the algorithm.

3.2 *Artificial Neural Networks (ANN)*

An Artificial Neural Network (ANN) is a computational model designed with the inspiration from human brain. They are widely popular and successful in both regression and classification tasks. The structure of ANN mimics the communication between neurons in the brain. Networks are built from multiple layers of neurons, and neurons in each layer connect by sending and receiving signals like the transmission across synapses in the biological brain. The input layer gets the data and sends it across the hidden layers till the output. The number of hidden layers determines how deep the neural network is. This method which allows multi-layers of abstraction, is called "Deep Learning" (LeCun, Bengio, & Hinton, 2015). The complex structure of ANN allows an infinite number of different combinations when building the model. Therefore, we will introduce some of the features chosen in the design.

Optimizers are algorithms or techniques that are used to minimize the loss function. They use different strategies to adjust the weights and learning rate during backpropagation (Carvalho, Lourenço, & Machado, 2021). For this task, we chose "Adam" optimization algorithm after some experiments. Adam optimizer is a well-known and widely used gradient descent optimization technique. It's a technique for determining individual adaptive learning rates for each parameter. Similar to Adadelta and RMS-Prop, It keeps both the moving average of previous gradients and the moving average of past squared gradients. As a result, it incorporates the benefits of both strategies (Kingma & Ba, 2014).

Neural network models have complex structures, and that makes them prone to overfitting. There are a couple of regularization techniques to

tackle this issue. We used Dropout and L1&L2 regularization. In dropout, some nodes are dropped or simply ignored at random during the training. This inhibits units from over-co-adapting (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). L1&L2 regularization methods use a different approach to reduce overfitting. They add an extra component, namely the regularization term, to the loss function. The values of weight matrices drop when the regularization term is included. It is assumed that lower weight matrices in a neural network result in a simpler model, and simpler models will minimize overfitting. The difference between L1 and L2 regularizations is the penalty term. L1 adds the sum of the squared weights, and L2 adds the sum of the absolute weights of coefficients. Thus, L1 can reduce coefficients to zero, but L2 has a tendency to decrease coefficients equally. Since we don't have a large set of features in our dataset, L2 regularization seems a better option for our model.

In our dataset, we have mainly two different sets of features: lyrics and audio descriptors. Although both are relevant for our task, they represent different types of data. Therefore, the joint objective model stands out when we want our model to learn from both datasets. As Z. Zhang, Huang, Huang, Zhang, and Li (2019) stated, joint learning extracts correlated information across several datasets to dramatically increase the quality of trained networks instead of learning a single network from each dataset separately. It also could be helpful in case of data scarcity since it enables using multiple datasets simultaneously. In practice, two different networks can be modeled with the varying size of hidden layers and merged with a shared layer. Thus, it allows adjusting the complexity of networks for both data types to improve the overall performance. Although we have a relatively small dataset, these advantages were effective in choosing neural networks to train. So, we made our model learn from both features simultaneously and were able to show if it improves prediction performance.

3.3 *Random Forest*

The Random Forest algorithm was introduced by Breiman (2001). It is a supervised machine learning technique used for regression and classification tasks. As the name suggests, a random forest represents the collection of decision trees. Each decision tree in the forest learns from a randomly selected sample of observations in the dataset. Additionally, a random subset of features is chosen to grow a tree at each node. For the output, the model takes the average for regression and the majority for classification. As a result, there is a lot of variety, leading to a superior model.

The notion behind the random forest algorithm can be easily recognized with a real-life example. Imagine you decided to buy a car and seek advice from your friends. The first friend asked about your budget and the size of your family. She recommended you a model based on your responses. This is the common strategy of the decision tree algorithm. Using your replies, your friend devised guidelines to help you decide which car you should buy. Following that, you began to seek even more friends for advice, and they responded by asking you a series of questions from which they may deduce some recommendations. Eventually, you selected the car that is most recommended to you, as in a random forest algorithm.

One of the biggest reasons we chose the random forest algorithm as a second model in our training is the "feature importance" attribute. We preferred the random forest regressor offered by the Sklearn library. Sklearn library provides a tool for measuring the relevance of a feature by calculating the mean and standard deviation of summed impurity decrease for each tree. It scales the outcomes so that the total significance of all factors equals one. Before training, numerous hyper-parameters can be modified. Some of the essential hyperparameters will be described here since hyper-parameters tweaking is a crucial component of the execution and provides insight into the algorithm's inner functioning. The hyper-parameters can be categorized according to their intended use. They are either employed to improve the model's predictive power or to help it run faster. We will focus on the first part.

n_estimators: Numbers of trees

A random forest, as previously stated, is the formation of several decision trees, also known as estimators. **n_estimators** hyper-parameter allows specifying how many decision trees will be constructed. Higher numbers of trees can improve the model's predictive power as it can also be computationally expensive.

max_features: Maximum number of features

This hyper-parameter limits the number of features that a random forest assesses when splitting a node. Increasing the number of features can boost the model performance since there will be a larger number of alternatives to examine at each node. On the other hand, higher numbers will diminish the diversity between trees and slow the computation.

min_sample_split: The minimum number of samples for split

This number decides how many observations are expected in each node to split. With more splits, the tree could incline to overfit. The minimum number of samples can be raised in order to tackle this

min_sample_leaf: The minimum number of samples in the leaf node

A decision tree's final node is called a leaf. The minimum number of samples required at a leaf node is specified by **min_sample_leaf**. The mod-

Variable	Description
#	unique song number
Country	name of the participant country
#.1	entry number of the country
Artist	name of singer(s)
Song	name of the song
Language	language(s) of the song
Pl.	place of the country
Sc.	score of the country
Eurovision_Number	number of the organization
Year	year of the organization
Host_Country	country of the organization
Host_City	city of the organization
Lyrics	original lyrics
Lyrics translation	English translation of lyrics

Table 1: Variables and their descriptions from the Eurovision lyrics dataset

els with the smaller leaves could be more sensitive to noise in the training set since it causes creating a separate path for almost every observation.

max_depth: Maximum depth of the tree

This hyper-parameter determines the longest route between the root and the leaf node. Increasing the depth helps the model learn more from training data, but at some point, it also could be overfitting since the outputs are no more generalizable for the out-of-training.

4 EXPERIMENTAL SETUP

4.1 *Datasets*

Two datasets were used for this thesis project. The first dataset stored in JSON format was downloaded from Kaggle.com under the name of "Eurovision Song Lyrics" (Minitree, 2021). The Eurovision Song Lyrics dataset contains lyrics and entry metadata of songs performed in Eurovision from 1956 to 2021 (Spotify, n.d.). In Total, it has 1644 songs. Each song has 14 features which can be seen in table 1. The lyrics are represented in their original language and translated version. All non-English lyrics had been translated to English and stored as a separate feature. If their original language is English, then the lyrics translation feature only includes "English," not original lyrics again. In addition, there are two empty translations for the lyrics that are created from an imaginary language.

The second dataset consists of the data extracted via Spotify Web API. There are several playlists that aim to combine all the Eurovision songs available on Spotify. As a result of our search, we used the songs in the playlist called Eurovision (1956-2021) | Eurovision Song Contest (1956-2021. In Total, the dataset consists of 1384 songs with 19 different elements. Each track contains the information of Playlist ID, Track Name, Track ID, Sample URL, Release Year, Genres, Duration, and Popularity in addition to the audio features, which can be seen in Appendix A (page 31). All audio-related features are continuous variables except key and mode that are categorical. All songs share the same playlist ID since they were downloaded from the same playlist, but each track has a unique Track ID. A track's popularity is measured on a scale of 0 to 100. According to Spotify, it is determined by an algorithm that considers the total number of listens and their recency.

4.2 *Exploratory Data Analysis*

Exploratory Data Analysis (EDA) was used to investigate the dataset in greater depth. Figure 2 shows the frequency distribution of normalized scores. The figure shows that the distribution is asymmetric and skewed to the right except for the highest scores. It is highly common to leave the competition with very low scores. The score difference between top and bottom songs is quite high. It can be derived that actually, a few songs are competing for the winning position whereas other songs are not able to collect many points.

Figure 3 shows the frequency distribution of word counts in lyrics. At first glance, the distribution seems close to normal. The average word count is 222. It is worth mentioning that this number could slightly change since we are analyzing the translated lyrics. A few lyrics have a very high or low word count. In our dataset, the lowest word count is 30, which belongs to Norway, the winner in the same year. We also created a word cloud to find the frequently used words in the lyrics. As can be seen in figure 4, love is a common theme in the songs. Words "heart" and "love" were repeatedly used over the years. It is also noteworthy that catchy phrases such as "oh oh" and "la la" are frequently used in the lyrics. It is probably due to making it easy for the audience to remember the song and sing along.

Finally, we created a correlation matrix to check the correlation coefficients between continuous audio features (see figure 5) . The first thing to notice in the table is the high negative correlation between acousticness and energy. Matrix also implies a strong positive correlation between energy

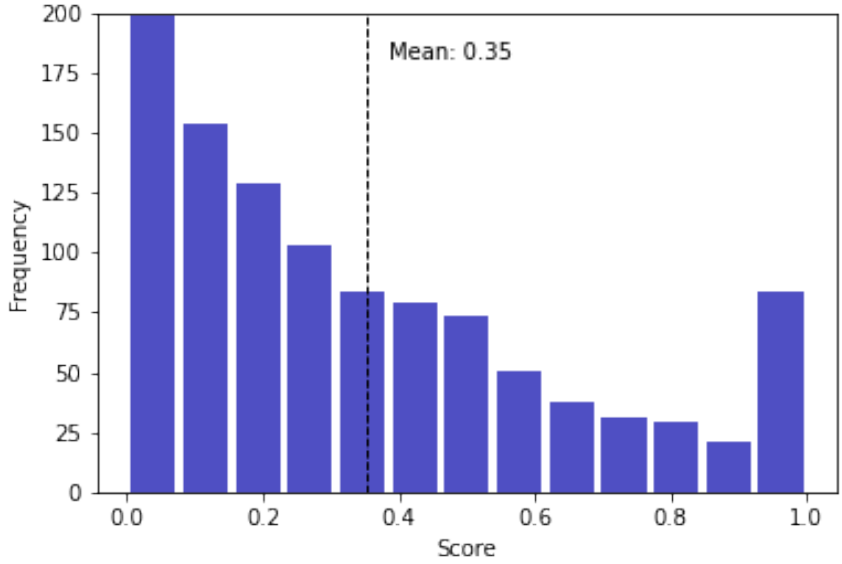

Figure 2: Frequency distribution of (normalized) scores in Eurovision

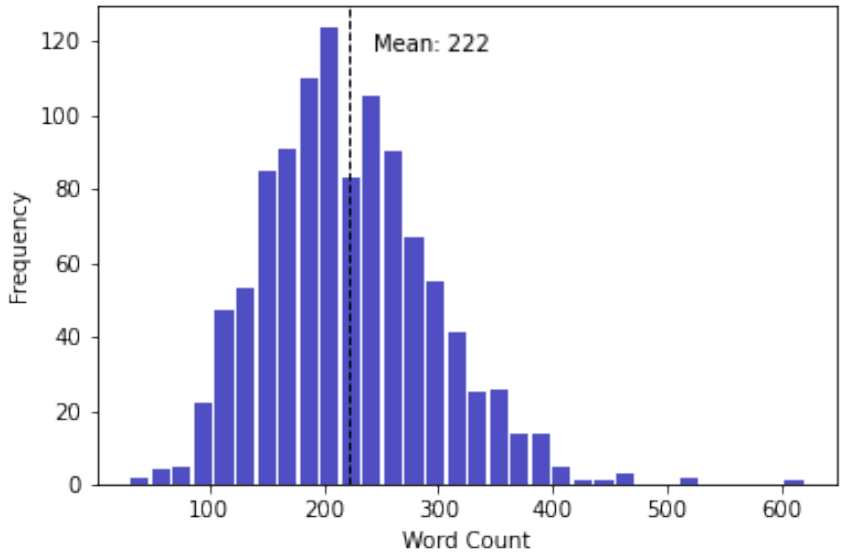

Figure 3: Frequency distribution of word counts in lyrics

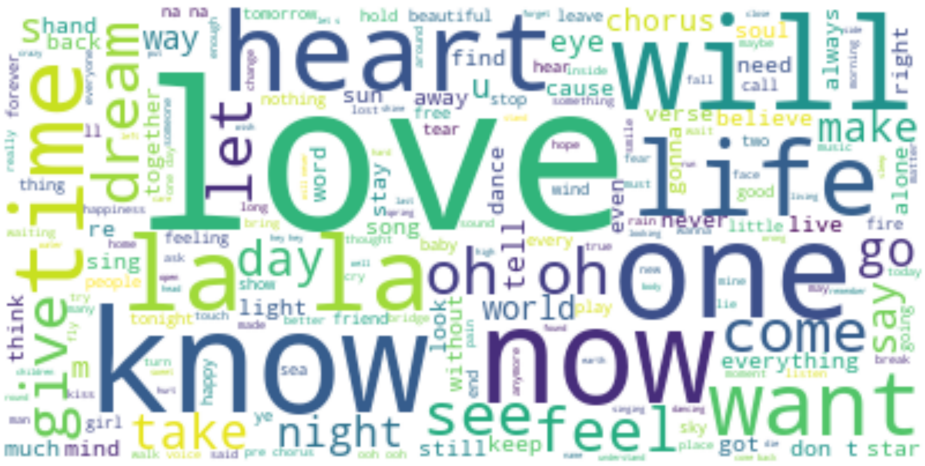

Figure 4: Graphical representation of word frequency in (translated) lyrics. The larger the term in the image, the more frequently it appeared in lyrics.

and loudness. These findings seem expected since, in general, acoustic songs are less energetic, and songs with a lot of energy are loud and noisy.

4.3 *Pre-processing*

The first task was to merge two datasets into one. Since the names of the songs don't always match one hundred percent, and there are multiple songs with the same name; manual intervention was necessary. The Eurovision Song Lyrics dataset is complete, so we checked all pieces there with the extracted data. We verified with other related information or listening on Spotify in case of any confusion, such as the same song names. When the song is identified, we added the unique Track ID to the lyrics dataset. At the end of the procedure, we merged two datasets on Track ID column.

Original English lyrics were copied to the translated lyrics column since we used only English-translated ones. All columns were converted to the best possible data types. We had to drop some rows for various reasons. Firstly, we dropped the rows when the "Pl." column was null. In the recent years of the competition, also the semi-finals started to be held. Countries have a place if only they performed in the final. Since the score in the semi-finals and finals are not compatible, we only used the songs performed in finals. A particular situation applies to the first year of the competition. In 1956, only the winning country was announced, so there was no place for the rest of the competitors. Secondly, we dropped the songs without audio features since not all Eurovision songs are available on

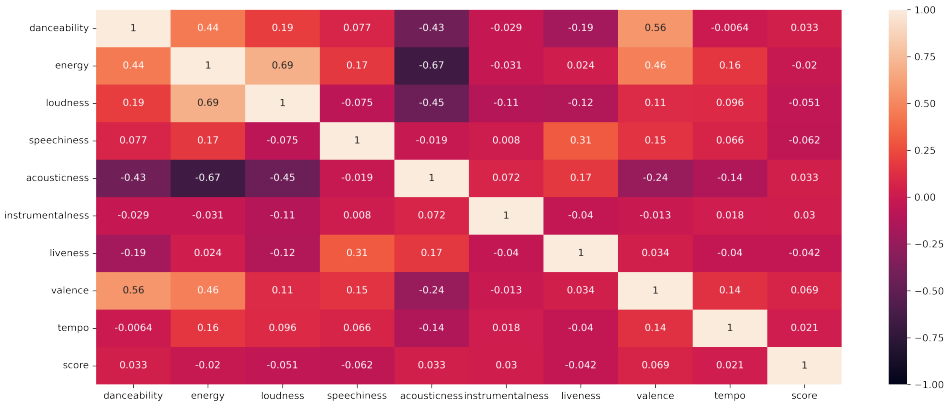

Figure 5: Visualization of correlation between continuous audio features as a matrix. The color indicates the intensity of correlation as shown in the color bar on the right side.

Spotify. Lastly, we dropped two songs with imaginary languages because it is not sensible to use them for our lyrics model. In the end, we had 1076 songs.

The data transformation was also implemented on some columns. Our dependent variable is the score of a country in the competition. The scoring system has been changed several times, and each year has a different scale of scores. That's why we applied min-max normalization for the scores within the same year. Thus, the scores had a range of 0 to 1. Normalization was also applied to the audio features if their original range is not 0 to 1. Furthermore, the categorical feature, key, was one-hot encoded. Finally, we transformed the translated lyrics to embeddings. Translated lyrics were used instead of originals because the range of used languages is pretty diverse, and some songs even consist of more than two languages. The transformation was implemented with BERT. First, text inputs were normalized in the "uncased" manner, which means that the text was lower-cased before being tokenized into word fragments, and any accent indicators were removed. Then, it generated an output embedding size of 1024 for each lyric.

The data was split into training and test set as 90 percent and 10 percent, respectively.For validation, we applied validation split for neural network and cross-validation for random forest models on the training set. Detailed adjustments will be described further.

4.4 *Models*

4.4.1 *Artificial Neural Network*

The first feed-forward neural network was created for audio descriptors. We trained our model with the data extracted from Spotify. Only the audio features were used for this task. For the model design, refer figure 6. Based on our experiments, we used tanh as the activation function for the three hidden layers. In the output layer, the activation function was sigmoid. We preferred sigmoid over linear because our output variable is in the range of 0 to 1 as in the sigmoid function's expected return range. To tackle overfitting, L2 regularization was applied on each hidden layer with a rate of 0.0001, and dropout was utilized. Chosen optimizer was Adam, and the initial learning rate was 0.0001. Finally, we decided MSE as our loss function and fitted the model with a batch size of 32 for 1000 epochs.

The second model was trained with BERT embeddings. For the design of the model, refer to figure 7. The lyric model is a deeper version of our initial model for the audio. It has two additional layers. The reason behind this is that the input size,1024, is larger compared to 22. Tanh was used as the activation function for the five hidden layers. In the output layer, the activation function was sigmoid. At a rate of 0.0001, L2 regularization was applied to each hidden layer. Adam was chosen as the optimizer, and the starting learning rate was 0.0001. Finally, we chose MSE as our loss function and fitted the model for 1500 epochs with a batch size of 32.

For the last model, we joined modified versions of the first and second models. You can see the detailed visualization in figure 8. Again, the activation function is sigmoid for the output, and hidden layers have tanh activation. L2 regularization was applied to shared hidden layers at a rate of 0.001. The optimizer was chosen as Adam, and the learning rate was set to 0.0001. Finally, we used MSE as our loss function and fitted the model for 1000 epochs with a 32-batch size.

For all three models, the validation split was adjusted as 20 percent.

4.4.2 *Random Forest*

For lyrics and audio features, also the random forest algorithm was trained. We didn't apply a combined model for this part because the random forest is not suitable for a joint model like a neural network.

In order to find optimal hyper-parameters, the Grid Search Cross-Validation from the Sklearn library was applied. There were three folds for each of the 288 candidates for the audio model, resulting in a total of 864 fits. Candidate hyper-parameters are as follows:

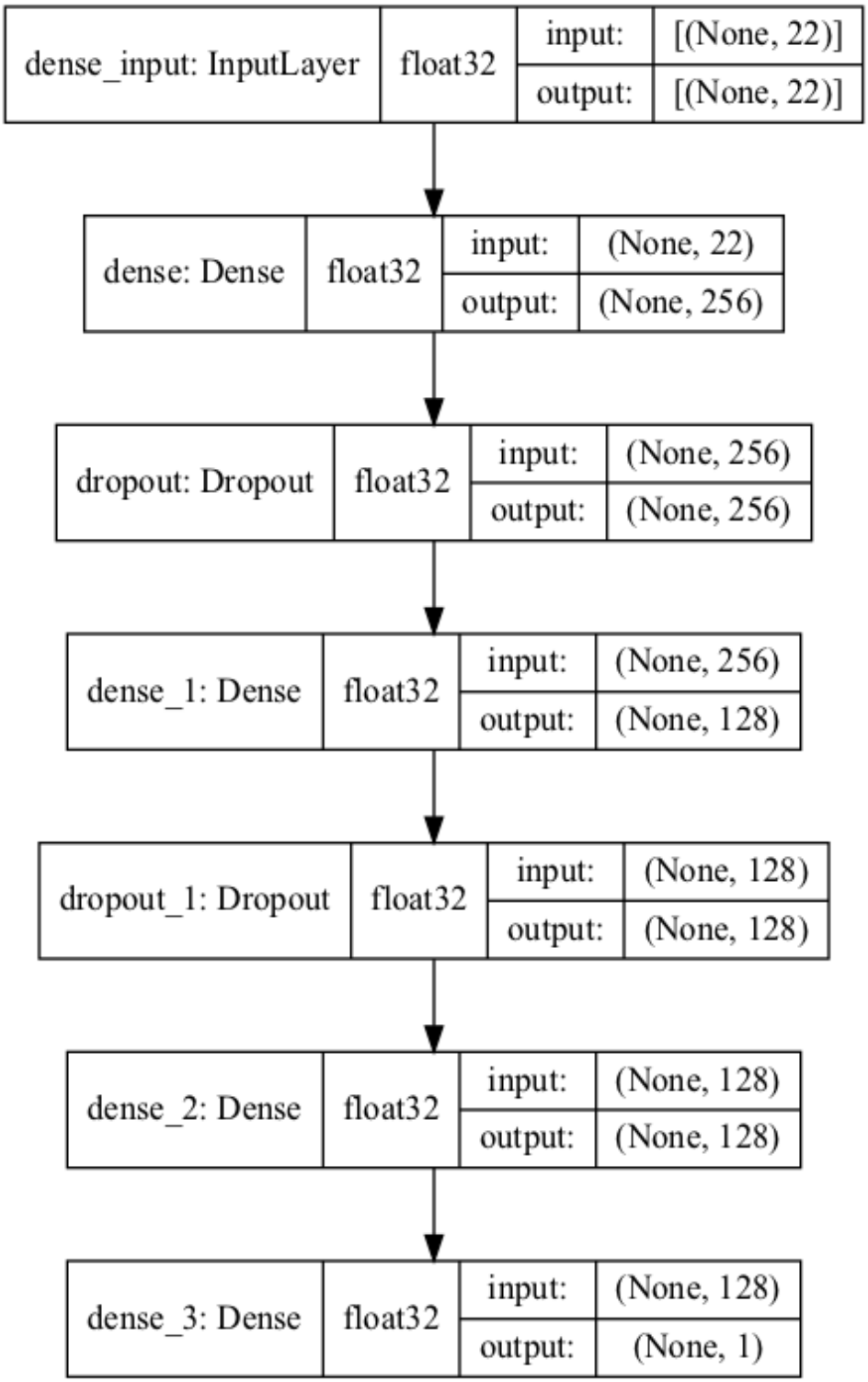

Figure 6: Graphical representation of ANN model for audio features

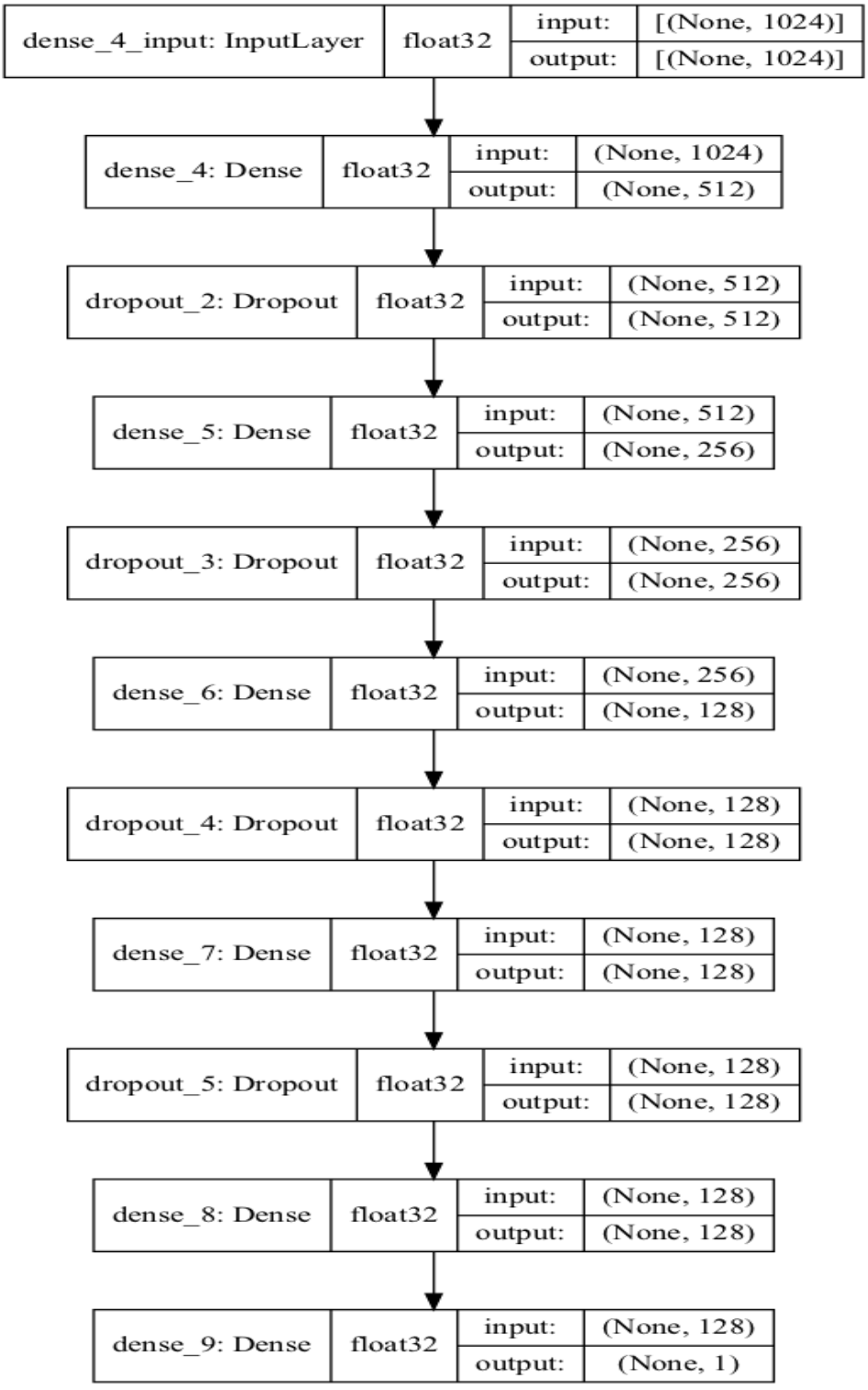

Figure 7: Graphical representation of ANN model for lyrics

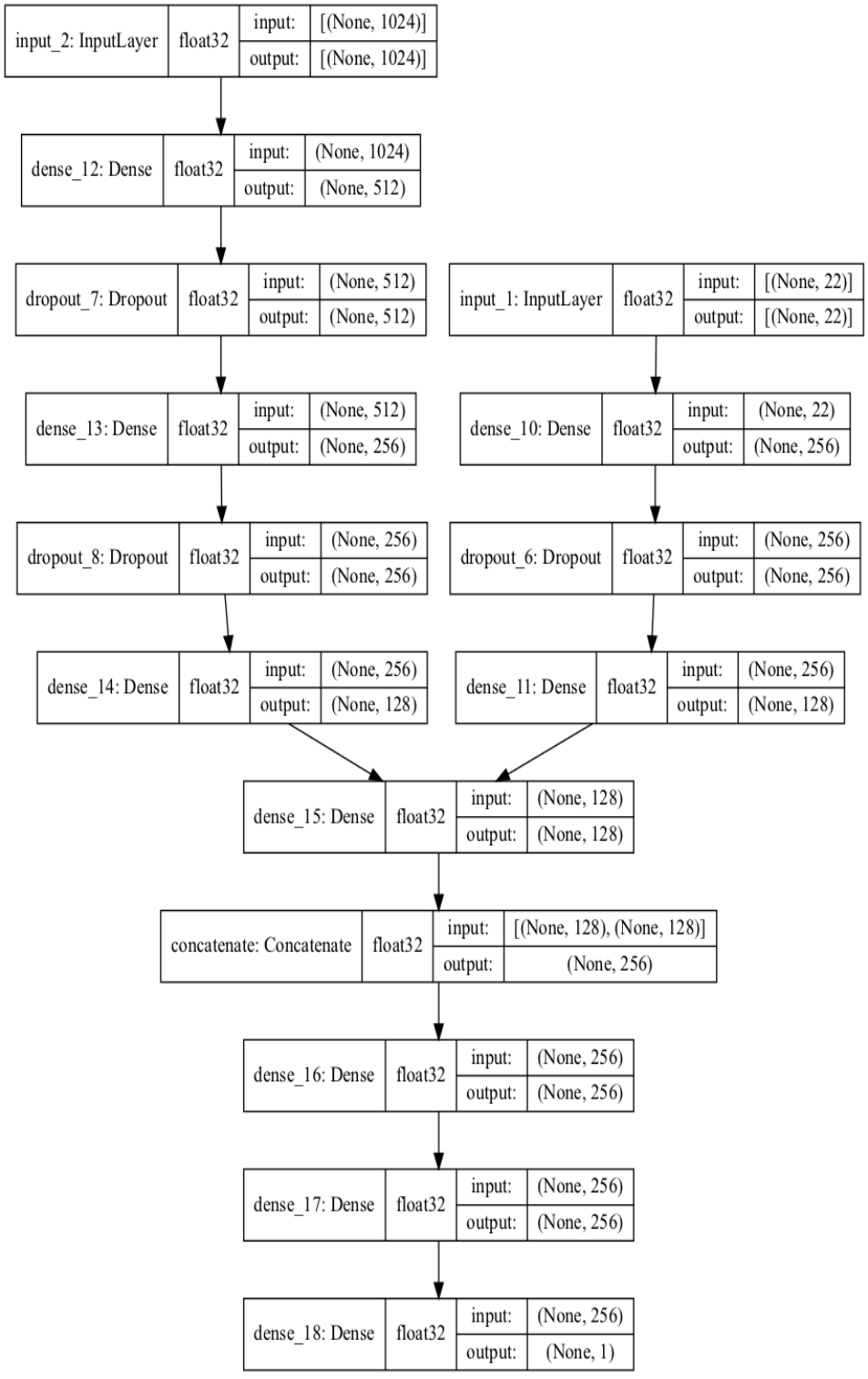

Figure 8: Graphical representation of joint ANN model for audio features and lyrics

param_grid = {
'max_depth': [80, 90, 100, 110],
'max_features': [2, 3],
'min_samples_leaf': [3, 4, 5],
'min_samples_split': [8, 10, 12],
'n_estimators': [100, 200, 300, 1000]
}
Chosen hyper-parameters for training our audio model : {
'max_depth': 80
'max_features': 3,
'min_samples_leaf': 5,
'min_samples_split':12,
'n_estimators': 100
}

The same procedure was also followed for lyrics. There were three folds for each of the 1920 candidates, resulting in a total of 5760 fits. Candidate hyper-parameters are as follows:

param_grid = {
'max_depth': [80, 90, 100, 110, 120],
'max_features': [2, 3, 5, 7],
'min_samples_leaf': [3, 4, 5, 7],
'min_samples_split': [8, 10, 12, 14],
'n_estimators': [100, 200, 300, 400, 500, 600]
}
Chosen hyper-parameters for training our lyrics model : {
'max_depth': 100
'max_features': 2,
'min_samples_leaf': 7,
'min_samples_split':8,
'n_estimators': 100
}

Lastly, we decided on the baseline model. Since the score ranges from 0 to 1 after normalization, we calculated evaluation metrics by giving 0.5 to every song.

4.5 *Software*

Python 3.8.5 was employed as the programming language. The pre-processing and analysis were performed in the Jupyter Notebook (6.4.5) using Anaconda Navigator 2.1. This study made use of a variety of libraries and packages, including the Keras, TensorFlow libraries for neural networks and BERT embeddings, skicit-learn (sklearn) for machine learning

tasks, Matplotlib for visualizations, and for other operations NumPy and Pandas.

4.6 *Evaluation Metrics*

For this regression task, we will evaluate and compare our models with three metrics:

Mean Absolute Error (MAE) :

The average of the absolute difference between actual and predicted values is calculated using the MAE measure.

$$\frac{1}{n}\sum_{t=1}^{n}|e_t| \tag{1}$$

MAE is more resistant to data that contains outliers. As a disadvantage, MAE is not a differentiable function, making it difficult to perform mathematical calculations.

Mean Squared Error (MSE) :

MSE is a widely used and simple metric with a small difference from MAE. Compared to MAE taking the absolute, MSE takes the squared difference between the actual and predicted value and averages it.

$$\frac{1}{n}\sum_{t=1}^{n}e_t^2 \tag{2}$$

It can easily be used as a loss function in contrast to MAE, but it is more sensitive to outliers since it takes the square of the difference.

Root Mean Squared Error (RMSE)

The formula of RMSE is quite intuitive. It is just the root of MSE.

$$\sqrt{\frac{1}{n}\sum_{t=1}^{n}e_t^2} \tag{3}$$

Because it has the same units as the output variable, RMSE is more often employed than MSE to assess the performance of the regression model.

5 RESULTS

We trained five models by applying the procedure described in section 4. They can be grouped as the audio models, lyric models, and audio + lyric model. Both neural network and random forest were applied for lyric and audio, whereas only neural network was trained for the combined model. In this section, the performance of each model will be presented along with the baseline score. Refer table 2 for a comparison of evaluation metrics.

	Audio			Lyrics			Audio + Lyrics		
	MAE	MSE	RMSE	MAE	MSE	RMSE	MAE	MSE	RMSE
Baseline Model	0.301	0.113	0.337	0.301	0.113	0.337	0.301	0.113	0.337
Neural Network	0.229	0.077	0.278	0.230	0.078	0.280	0.206	0.071	0.266
Random Forest	**0.214**	**0.067**	**0.259**	0.226	0.073	0.270			

Table 2: Evaluation of each model on test set

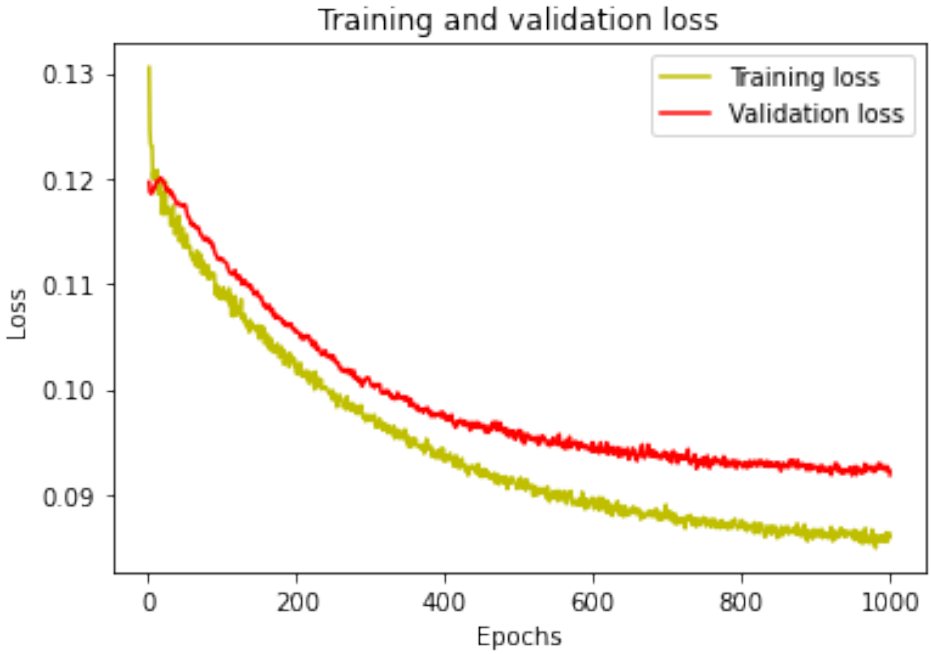

Figure 9: MSE losses of ANN audio model during training

5.1 *Audio Model*

The first audio model was trained with the neural network; see figure 9 for the loss profile. It took 1000 epochs to train our neural network model for score prediction using audio descriptors. The loss profile shows that validation loss stopped decaying after 800 epochs and then stabilized. However, training loss continued to decrease slowly. As mentioned before, we applied dropout and kernel regularization to cope with overfitting after multiple trials. Also, we adjusted the learning to a lower rate. These modifications can explain the slow but smooth trend.

The second model was the random forest. Both random forest and neural network models yielded more minor losses compared to baseline (0.337 RMSE). The neural network scored 0.278 RMSE on test data. However, the random forest had the best performance with 0.259 RMSE. Additionally,

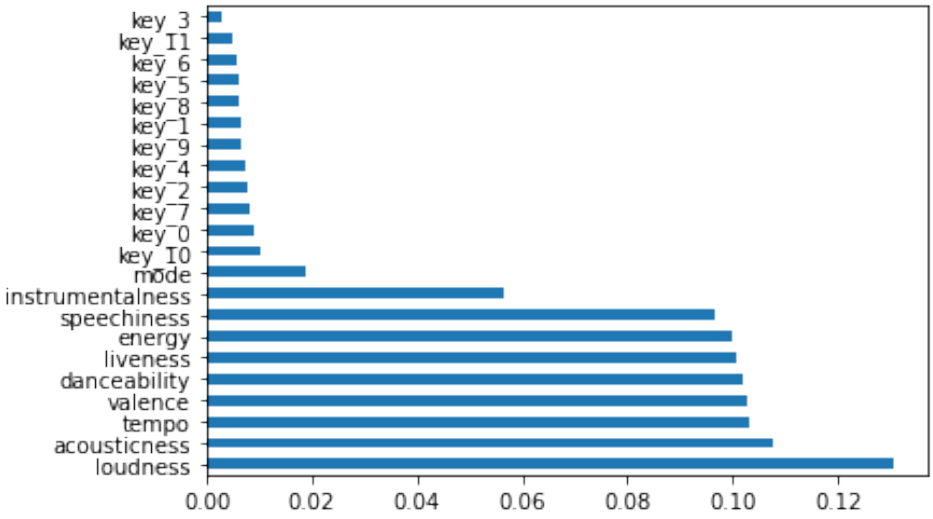

Figure 10: Feature importance of audio features in random forest model

the lowest loss was achieved with the random forest audio model among all the models. It can also be analyzed that to what extent each audio feature contributed to the model; see figure 10 for the feature importance in trained random forest algorithm. It shows that the most relevant feature for the prediction task is loudness. This outcome supports prior research in literature. According to PINARBAŞI (2019), loudness is the foremost important classifier for songs in terms of popularity ratings. In our model, acousticness, tempo, valence, and danceability follow it. However, all continuous features except loudness reduce the impurity of splits at similar degrees. In other words, they have similar importance for the model.

5.2 *Lyrics Model*

Lyrics embeddings were used as input to train both models. Our neural network model lasted 1500 epochs to train. The loss curve was relatively smooth as in the audio model. At around the 800th epoch, the training and validation loss converged (see figure 11). Afterward, training loss followed the downtrend, while validation loss started to increase. After convergence, fluctuations raised in validation loss, as could be seen as the result of overfitting, but the overall trend was maintained.

Again, both models performed better compared to baseline. However, the difference between the two models was not significant. RMSE was 0.28 for the neural network and 0.27 for the random forest. Both losses were higher than the audio models.

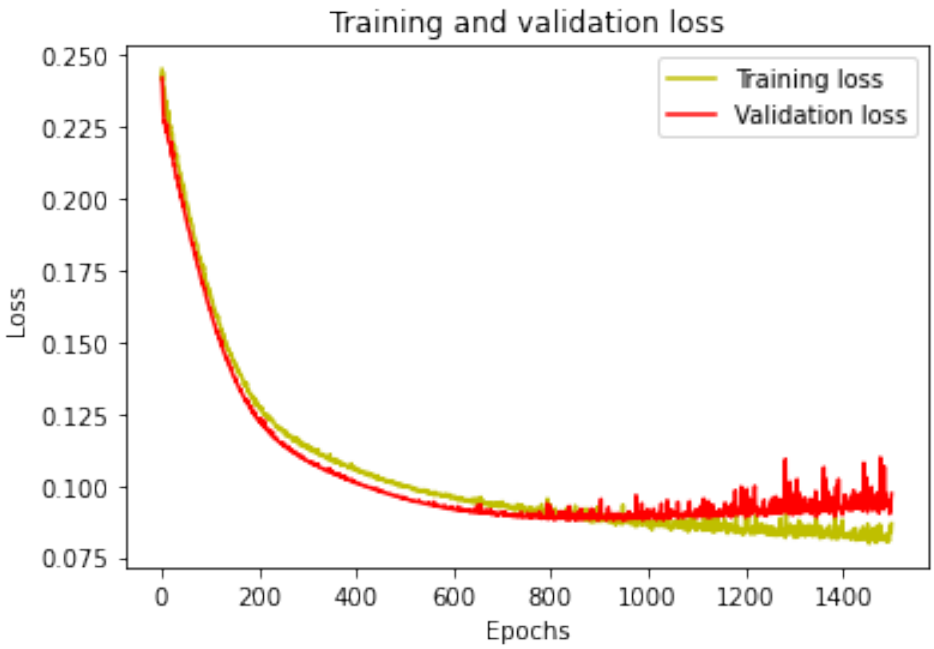

Figure 11: MSE losses of ANN lyrics model during training

5.3 *Audio + Lyrics Model*

Lastly, we combined both neural networks for audio and lyrics into one. We merged two models with some alterations, but both architecture mostly stayed the same. The final model was trained over 1000 epochs. As seen in figure 12, after around 200 epochs, loss for validation stopped decreasing. Then, it plateaued for a while and started increasing, whereas training loss continued to decline but at a slower pace. RMSE score was 0.266 on test data which is the lowest compared to audio and lyric neural network models. Also, the combined model performed better than the baseline. Findings support the idea that integrating two models as a joint objective model will increase the model's predictive power. Nevertheless, the performance of the joint model didn't pass the random forest. The combined model is the second-best model, following the random forest trained with audio features.

6 DISCUSSION

This thesis aims to explore whether it is possible to predict a score of a song performed in Eurovision and, if so, to what degree. Consequently, we plan to shed light on which characteristics of a piece contribute to its

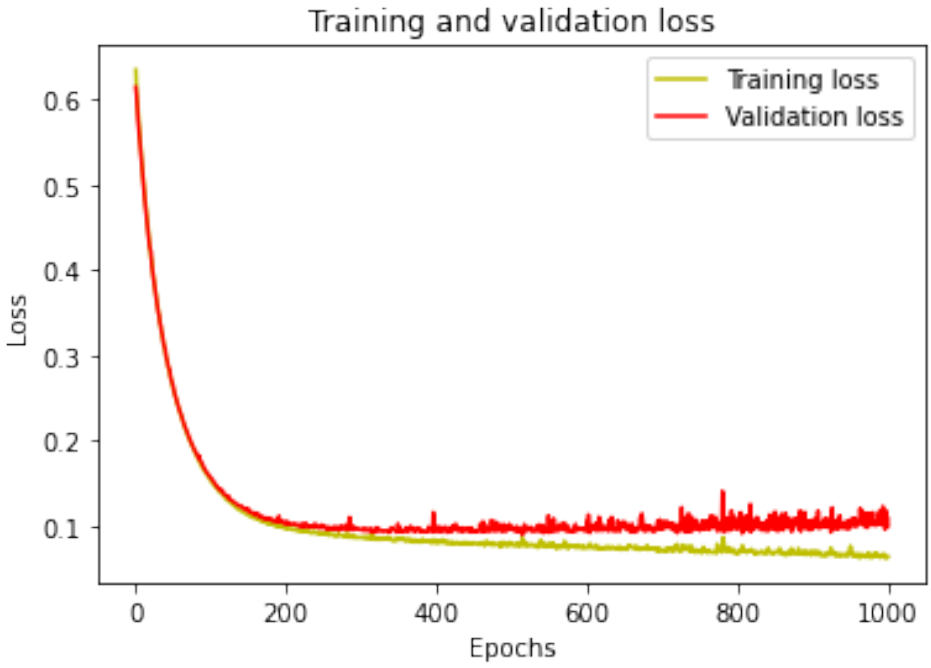

Figure 12: MSE losses of ANN audio + lyrics model during training

success or failure in the competition. In accordance with this purpose, we categorized internal factors of a song as lyric and audio features. Later, we trained our models in a way as our related sub-questions pointed.

The first sub-question was if audio descriptors can help predict the score of a song. Random forest and neural network models were trained to answer this question. The initial conclusion is that both models outperformed the baseline. This outcome can be interpreted as that both models are capable of learning from audio features to some extent. It is also possible to compare two models' performance in this task. Random forest more successfully predicted the score than the neural network. The earlier findings of Middlebrook and Sheik (2019) suggest the same result regarding the random forest's success over neural network. When we analyzed the feature importance in random forest's design, the loudness is significantly more effective in predicting the score. The interesting finding is that most of the other audio elements have a similar degree of importance for the model. We did not normalize most of the audio features since they are already in the range of 0 and 1; it could have affected the outcome because some of the audio features, such as instrumentalness and speechiness, are more likely to be represented with very small numbers although in the desired range. Normalization could have been more helpful in representing this distribution, or this outcome could also reflect the actual situation.

The second sub-question was if the lyric of a song can help predict its score. This time, we replicated the procedure with lyrics. Again, both models were better than baseline. However, lyric models performed worse than audio models. In literature, the comparison of lyrics and audio features is a relatively unexplored topic. Still, it is interesting that our finding is not aligned with the previous research (Demetriou et al., 2018; Dhanaraj & Logan, 2005) . They had pointed out the lyrics as a more important factor. Using original lyrics instead of translations could have yielded better performance. But then we would have to remove some songs from our dataset. We made this decision in order not to shrink our already small set. Another explanation of why lyrics models did not perform better could be the transformation of lyrics. With the help of BERT, context-based word embeddings were used in the training of lyrics models, and it is known that contextual embeddings are very successful in the representation of text inputs. Nevertheless, other text representation techniques can be tried for further research since it could also be claimed that lyrics mostly consist of verses and are not very dependent on context. Furthermore, other lyric features can be used for better performance instead of text representation. Initially, we thought about extracting additional information from lyrics such as emotion and rhyme, but in our understanding, it was not coherent to combine different types of representations for training lyric models. However, it is still possible to train models with different features of lyrics for further research. Both models' performances were almost the same. Although we had a larger input size for lyrics, it did not help the neural network model achieve a lesser loss than the random forest since it is known that neural network models could be prone to require larger data for better performance.

The last sub-question examines the performance of the joint model. We hypothesized that combining audio and lyric models could improve overall performance. The findings support the idea. The best performing neural network model was the joined model. We can interpret this as the model learns how the lyrics and audio feature contribute to the score simultaneously. While we were combining, we tried to preserve the initial models. However, the neural network models are very suitable for hyper-parameter tuning. Therefore, better performance can be achieved with further changes.

Some points are relevant for multiple sub-questions. The first point is that all three models are better performers than baseline. However, the decrease in loss is minimal. In our understanding, the size of the dataset is the possible reason behind it. For the Eurovision dataset, there are not many options to increase the data points for training. As explained in the Experimental Setup section, we took all the necessary decisions to obtain

as many data points as possible. The size of the dataset could also be the reason why random forest had more predictive power than the neural network model. Thus, especially for the audio features dataset, future research should focus more on simpler models.

Overall, this study shows that lyrics and audio features are not yet strong predictors for a country's score in the competition. This result was not surprising since we were already aware that other factors such as politics and performance on the stage are also important for a contestant's success. However, the outcome is promising for further research in this field. For instance, we showed that the loudness of a song could be an important factor to predict the score. Therefore, stakeholders, including producers and songwriters, could consider these findings when making their songs.

7 CONCLUSION

In this thesis, we investigated how the lyrics and audio features of a song affect the score in the Eurovision song contest. We used English translations of the lyrics and audio characteristics retrieved using the Spotify Web API for this work.

To reiterate, the thesis's research question, as stated in the introduction, is: *"Can we predict the score of a song at the Eurovision Song Contest based on its lyrics and audio features?"*. Three sub-questions have been developed to answer this question, and they will be addressed further below.

In order to answer the set of questions, five different models were constructed. Firstly, we trained random forest and neural network models with BERT based lyrics embeddings. Later we introduced two models with audio features, and lastly, we used both lyrics and audio features to train a neural network model. We will try to answer the corresponding sub-questions by utilizing the outcome of these models.

> *RQ1. Can we predict the score based on audio features, if so, to what extent?*

We found that the audio characteristics affect the score obtained in the competition because random forest and neural network models performed better than baseline. Random forest's performance was also better than the neural network model. Furthermore, we discovered that loudness is the most important audio feature affecting the score. Nevertheless, the performance difference between our models and baseline was minimal, implying a lesser effect of a song's features on the score.

> *RQ2. Can we predict the score based on lyrics, if so, to what extent?*

Again, both random forest and neural network models outperformed the baseline. It shows that lyrics affect the score of a song. However, it is hard to say that the trained models were significantly better than the baseline. The effect of lyrics on score seems minimal.

> *RQ3. Can we predict the score based on the combination of lyrics and audio features, if so, to what extent?*

Combining lyrics and audio features improved the neural network model's performance. Therefore, it can be concluded that lyrics and audio features, combined, have a more significant impact on a song's success. Nonetheless, as compared to the baseline, this effect is minor.

To sum up, both audio and lyrics characteristics of a song have the potential to influence the success in Eurovision. To the best of our knowledge, this study is the first attempt to predict the score by using lyrics and audio features. Since Eurovision data is a relatively new research topic, we would like to share some ideas regarding future research.

First, as stated previously, simpler models could be tried to deal with the small dataset. Secondly, other audio-related features can be added. In this study, we used audio descriptors provided by Spotify. However, audio features offered by other platforms could also give additional insights. Additionally, the problem can be treated as classification instead of a regression task by designating top and bottom songs (e.g., first five and last five songs), but it will probably decrease the size of the dataset. Another idea might be to combine this study with past research. For example, we already know that sentiment analysis of social media could be helpful in order to predict the success of a song. Combining internal factors of a song and the sentiment of social media towards it can drastically boost the model's performance. Lastly, but not least, original lyrics could be used for training, yet, as discussed before, it should be kept in mind that this could shrink the data size since not all the lyrics are suitable for multilingual embeddings.

To conclude, we showed that lyrics and audio features could be essential elements to succeed in the competition. Eurovision data is growing year by year, so still, there is plenty of room for future studies.

REFERENCES

Araujo, C. V. S., de Cristo, M. A. P., & Giusti, R. (2019). Predicting music popularity using music charts. In *2019 18th ieee international conference on machine learning and applications (icmla)* (pp. 859–864). IEEE.

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–32.

Carvalho, P., Lourenço, N., & Machado, P. (2021). Evolving Learning Rate Optimizers for Deep Neural Networks. *arXiv preprint arXiv:2103.12623*.

Demergis, D. (2019). Predicting Eurovision Song Contest Results by Interpreting the Tweets of Eurovision Fans. In *2019 sixth international conference on social networks analysis, management and security (snams)* (pp. 521–528). doi: 10.1109/SNAMS.2019.8931875

Demetriou, A. M., Jansson, A., Kumar, A., & Bittner, R. M. (2018). Vocals in Music Matter: the Relevance of Vocals in the Minds of Listeners. In *Ismir* (pp. 514–520).

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dhanaraj, R., & Logan, B. (2005). Automatic Prediction of Hit Songs. In *Ismir* (pp. 488–491). Citeseer.

Grabor, M. (2021). *Eurovision Song Contest Organizers Say 2021 Show Brought In 183 Million Viewers.* Retrieved from https://deadline.com/2021/05/eurovision-song-contest-2021-183-million-viewers-1234766929/

Kakouris, D., Theocharis, G., Vlastos, P., & Memon, N. (2016). Detecting hidden patterns in european song contest—eurovision 2014. In *Intelligent systems for computer modelling* (pp. 99–109). Springer.

Kim, S. T., & Oh, J. H. (2021). Music intelligence: Granular data and prediction of top ten hit songs. *Decision Support Systems*, *145*, 113535. Retrieved from https://www.sciencedirect.com/science/article/pii/S0167923621000452 doi: https://doi.org/10.1016/j.dss.2021.113535

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kumpulainen, I., Praks, E., Korhonen, T., Ni, A., Rissanen, V., & Vankka, J. (2020). Predicting Eurovision Song Contest Results Using Sentiment Analysis BT - Artificial Intelligence and Natural Language. In A. Filchenkov, J. Kauttonen, & L. Pivovarova (Eds.), (pp. 87–108). Cham: Springer International Publishing.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436–444.

Lee, J., & Lee, J.-S. (2018). Music popularity: Metrics, characteristics, and audio-based prediction. *IEEE Transactions on Multimedia*, *20*(11), 3173–3182.

Liu, Q., Kusner, M. J., & Blunsom, P. (2020). A survey on contextual embeddings. *arXiv preprint arXiv:2003.07278*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... Stoyanov, V.

(2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Middlebrook, K., & Sheik, K. (2019). Song hit prediction: Predicting billboard hits using spotify data. *arXiv preprint arXiv:1908.08609*.

Minitree. (2021). *Eurovision Song Lyrics.* Retrieved 2021-09-27, from https://www.kaggle.com/minitree/eurovision-song-lyrics

Ochoa, A., Hernández, A., Sánchez, J., Muñoz-Zavala, A., & Ponce, J. (2008). Determining the ranking of a new participant in eurovision using cultural algorithms and data mining. In *18th international conference on electronics, communications and computers (conielecomp 2008)* (pp. 47–52). IEEE.

Peters, M. E., Ruder, S., & Smith, N. A. (2019). To tune or not to tune? adapting pretrained representations to diverse tasks. *arXiv preprint arXiv:1903.05987*.

PINARBAŞI, F. (2019). Demystifying Musical Preferences At Turkish Music Market Through Audio Features Of Spotify Charts. *Turkish Journal of Marketing*, *4*(3), 264–279.

Raza, A. H., & Nanath, K. (2020). Predicting a Hit Song with Machine Learning: Is there an apriori secret formula? In *2020 international conference on data science, artificial intelligence, and business analytics (databia)* (pp. 111–116). doi: 10.1109/DATABIA50434.2020.9190613

Sharma, I. (2020). *Hit song classification with audio descriptors and lyrics* (Unpublished master's thesis). Rutgers The State University of New Jersey, School of Graduate Studies.

Si, Y., Wang, J., Xu, H., & Roberts, K. (2019, nov). Enhancing clinical concept extraction with contextual embeddings. *Journal of the American Medical Informatics Association*, *26*(11), 1297–1304. Retrieved from https://doi.org/10.1093/jamia/ocz096 doi: 10.1093/jamia/ocz096

Spotify. (n.d.). *Eurovision (1956-2021) | Eurovision Song Contest (1956-2021.* Retrieved from https://open.spotify.com/playlist/39mNowYJVhim6TyP6IMipP?si=d650320c4e8f43bd

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), 1929–1958.

Stieglitz, S., Meske, C., Ross, B., & Mirbabaie, M. (2020). Going back in time to predict the future-the complex role of the data collection period in social media analytics. *Information Systems Frontiers*, *22*(2), 395–409.

Tenney, I., Das, D., & Pavlick, E. (2019). BERT rediscovers the classical NLP pipeline. *arXiv preprint arXiv:1905.05950*.

Yang, L., Chou, S., Liu, J., Yang, Y., & Chen, Y. (2017). Revisiting the problem of audio-based hit song prediction using convolutional neural networks. In *2017 ieee international conference on acoustics, speech and signal*

processing (icassp) (pp. 621–625). doi: 10.1109/ICASSP.2017.7952230

Yu, L.-C., Yang, Y.-H., Hung, Y.-N., & Chen, Y.-A. (2017). *Hit song prediction for pop music by siamese cnn with ranking loss.*

Zangerle, E., Pichl, M., Hupfauf, B., & Specht, G. (2016). Can Microblogs Predict Music Charts? An Analysis of the Relationship Between Nowplaying Tweets and Music Charts. In *Ismir* (pp. 365–371).

Zangerle, E., Vötter, M., Huber, R., & Yang, Y.-H. (2019). Hit Song Prediction: Leveraging Low-and High-Level Audio Features. In *Ismir* (pp. 319–326).

Zhang, T., Wu, F., Katiyar, A., Weinberger, K. Q., & Artzi, Y. (2020). Revisiting few-sample BERT fine-tuning. *arXiv preprint arXiv:2006.05987*.

Zhang, Z., Huang, X., Huang, Q., Zhang, X., & Li, Y. (2019). Joint Learning of Neural Networks via Iterative Reweighted Least Squares. In *Cvpr workshops* (pp. 18–26).

8 APPENDIX A

Variable	Description
Acousticness	A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
Danceability	Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.
Energy	Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity.Typically, energetic tracks feel fast, loud, and noisy. For example, death metal has high energy, while a Bach prelude scores low on the scale. Perceptual features contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy.
Instrumentalness	Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context.Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.
Key	The key the track is in. Integers map to pitches using standard Pitch Class notation.
Liveness	Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.
Loudness	The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track and are useful for comparing relative loudness of tracks. Loudness is the quality of a sound that is the primary psychological correlate of physical strength (amplitude). Values typically range between -60 and 0 db.
Mode	Mode indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is 0.
Speechiness	Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layerd, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks.
Tempo	The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.
Valence	A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).

Table 3: Description of audio features retrieved via Spotify