A Comparison of Classifiers for Rap Authorship Attribution

Thomas Horak, Brian Guo, Krishan Amin, Sawit Trisirisatayawong, Vraj Desai
Dept. of Computer Science
University of Michigan Ann Arbor
Ann Arbor, MI 48104
{thorames,brianguo,kmamin,sawittri,desaiv}@umich.edu

1. Introduction

Rap, whose origins trace back to ancient African storytellers, has transcended cultures and become an integral part of everyday life. Rappers have molded their lyrics to represent their unique geographic regions, race and environment. For the past few decades it has been possible to query artist lyrics and match them to their respective rappers, but being able to query a unique combination of words and return a most similar rapper unlocks another realm of discovery. Such a product allows the average person to see which rappers and eras their own lyrical style matches with and has potential to be used by musicians, researchers, and those interested in the evolution of rap.

Linguistically, rap is unique, as it does not follow grammar rules, and adheres to a different structure that's borders more poetic than prosodic. Nevertheless, it's far less codified and understood in its method and is generally highly variable across countries, cities, time periods. Our project aims to understand the intricacies of the language and provide similarity analysis diving deeper than simply words, but the underlying language and structure as well.

Our group is interested in exploring this topic primarily due to a recent resurgence of rap with vast stylistic differences. Modern rappers such as Cardi B and Logic both are well-known, respected rappers, but they both feature distinct styles that can be identified through their music and lyrics. According to USA Today, rap has taken over as the most popular music genre in the U.S, giving it a large cultural reach [1]. The younger population have especially grasped onto this genre through its relatable lyrics, catchy beats, and up-tempo rhythms.

Such an authorship attribution task involves many aspects — acquiring and preprocessing datasets, structuring similarity analyses, and finding unique relations with a given query. Before running queries, we must train our classifier with the preprocessed datasets in order to build language models representative of each rapper's discography. This training can be achieved using many different methods — it is our hope to test many separate methods of training, processing, and ranking such that we can analyze their differences and biases. We certainly do not expect our research to result in a state-of-the-art classifier for rap authorship attribution. Instead, we plan to take a more exploratory route, comparing and contrasting different methodologies in order to see which prove most promising for the future of this task.

Robust, complete datasets are integral to the success of this endeavor, so we are leveraging Genius' API, an industry leader in music lyrical content. Using this data along with NLP and IR strategies, we believe it possible to suggest similar artists to a user based on their lyrical input. We plan to utilize

many of the strategies we've learned throughout the course so far in addition to expanding our knowledge so as to explore new methods potentially capable of supplying interesting results. Our documentation will include a robust analysis of these findings and metrics on the success of all methods.

Ultimately, we hypothesize that in varying preprocessing techniques and feature usage, we will be able to determine which models and which configurations best function to solve the task of rap authorship attribution. Furthermore, we hope that as a result of our experiment we will be able to provide evidence which suggests that the computational classification of rappers is a promising field within NLP.

2. Scholarship

Text classification has been incredibly prevalent within machine learning and is a task fundamental to the field of natural language processing. Specifically, text classification is the process of assigning tags or categories to text according to its content. A classic, yet relevant example of text classification is the attribution of authorship to literary texts — it is fairly easy to implement a classifier which trains on the content of various texts (whether that be books, research publications, poetry, etc.) and makes predictions about who a written text's author is likely to be (one such example is displayed in the work of Sudheep Elayidom and his team [2]). It is fairly easy to imagine that if a classifier is capable of correctly assigning authors to their literary works, then it may also be the case that a classifier is capable of making predictions about which rap artist has produced a given lyric.

In recent years, it has been asserted that rap lyrics possess enough variation in their style and word use so as to be distinguishable between artists — for example, researchers have been able to map changing trends in drug use, alcohol consumption, and even the prevalence of violence depicted by the lyrics of rap artists solely based on their lyrical content [3] [4] [5]. The presence of such variation in word use and lyrical styling among artists' lyrics has allowed for a great deal of innovation in terms of the computational analysis of music. The evolution of rap lyrics throughout history has also been noted, from the 1990s where many artists focused on their difficult upbringings to modern rappers who often flaunt the riches and luxuries that characterize their lives [6].

Sam Isenberg, for example, has been able to create a classifier capable of assigning songs to a genre given only their lyrical content [7]. Isenberg's system is rather homogenous to what we hope to achieve with our system and likely suggests the viability of a classifier capable of correctly identifying a rap artist based on a given lyric. Another such example of computational analysis of music can be noted in work done by Anders Olson-Swanson — Olson-Swanson has been able to create an application which makes rapper recommendations to users. His system takes a rapper query provided by the user (namely, a rapper which the user particularly enjoys) and analyzes their lyrical content and word frequency so as to make recommendations of other rappers with similar word usage and lyrical styling [8].

Attempts have also been made to use audio data instead of textual information to classify songs and artists. Researchers at the University of Dortmund have built a classifier that takes 10 second audio clips and predicts their genre [9]. Relative to the other two classifiers built by Isenberg and Olson-Swanson, this audio based classifier proved to

be less accurate, which encouraged us to focus on lyrics as a more distinct identifier for each artist. In creating a text classification system for the attribution of authorship to rap lyrics, we hope to consider, learn from, and expound upon all of these aforementioned projects (and others of a similar nature), further proving that songs lyrics are useful in making computational judgements about artists and their musical works.

3. Data

In order to obtain data, we used the Genius API as a means for retrieving artist data, their songs, and song details. We began with a popularity based seedlist of 44 rappers to request from Genius. We requested by artist name and downloaded the 50 most popular songs for every artist in the seedlist. Following this, we formatted our data into a json dictionary which included information such as lyrics, artist, location, and year. This data is used by our classifier without any manual annotation.

4. Methods

Functionally, our project attempts to build and evaluate various classifier models capable of attributing authorship to rap lyrics. Various techniques in preprocessing and feature extraction were used to help elucidate which elements present in rap lyrics are truly important to the rapper classification task. We designed these elements and processes to take advantage of what we see as rap's unique characteristics and defining features.

Rap lyrics tend to often use very specific lexicons and syntactic structures — considering the linguist's perspective, rap uses an entirely different vernacular and follows its own rules, often not matching the expected semantic meaning for a language.

Rap's unique vocabulary led us to built a custom stopword set for our rap lyric corpus. This custom set of stopwords consists of the 50 most common used terms (longer than 4 characters in length) in our rap lyric corpus. This set is present within our system as *rapsw.txt* and is employed in the same manner as is a standard English stopwords set. Our motivation for the removal of this set of rap stopwords was our belief that these common terms were not necessary for semantic analysis due to their proliferation.

Overall, we attempt to perform a rigorous study which isolates the effects of preprocessing with lemmatization, stopwords, and use of our own custom set of rap stopwords. Furthermore, we allow for varying data representations, including Count Vectorization and Tfidf Vectorization, and take consideration of how part of speech counts may improve classification performance.

Our process involved the identification of promising models for this classification task. The models chosen were Multinomial Naïve Bayes, Decision Trees, Support Vector Machines, Locality Sensitive Hashing, Cosine Similarity, Latent Dirichlet Analysis (LDA), Latent Semantic Inference (LSI). LDA and LSI models were chosen because we believed that semantic topics would provide a reasonable way of distinguishing artists beyond simply vocabulary. Cosine Similarity with TF-IDF vectors, LSH and Naïve Bayes were chosen because of their simplicity and narrow focus on word use. Finally, Decision trees and SVMs were used to provide representation of more complex models.

5. Experiment

For each model, the same basic preprocessed corpus was built, where non-

alphanumeric characters and any sort of encoding characters were removed. Due to a lack of punctuation and common English syntactic rules in rap, base preprocessed tokenization was limited: no stopwords or punctuation was removed. Additionally, this corpus was the primary source of features for training each model, taking consideration of n-grams, part-of-speech counts (provided by the Spacy library), lyrical line length, and word length.

In order to train our models, we split our corpus via an 80-20% train-test split. We constructed a second version of the test set split where song lengths were ⅓ of their original size — this new testing dataset was used so as to determine which classifiers were capable of strong performance on shorter documents.

First, experimentation was done to determine each classifier's basic performance. Then, individual features and preprocessing techniques were altered in order to test for changes in performance. In particular, our experiments were structured by passing different combinations of preprocessing parameters -- this meant altering use of lemmatization, stopword removal, and rap stopword removal. This also included the inclusion and exclusion of features such as part of speech counts.

After determining the performance on the full length songs, the second corpus of snippet songs of ⅓ length was run with the models configured for highest performance. This means using the best combination of preprocessing and features that we found previously for the full length songs. Performance on these snipped length songs was also recorded for analysis.

6. Evaluation

Each model was trained on different combinations of the data, and then evaluated according to recall metrics to determine the effect of these different combinations. Our classification performance evaluation with a focus on recall @ k, for the values of 1, 5, 10. Recall, in this case, is determined simply by the presence of the correct author in the list of similar authors outputted by each classifier. Recall values for each query were averaged and then displayed as recall metrics for the entire model.

7. Model Configurations

Now, regarding how each model was individually trained and built. Each model's structure required slightly different inputs from the user beyond simply the corpus, and preprocessing, and some models were built from libraries.

For LDA and LSI, the models were built using the Gensim library. Both models constructed a number of topics, which we arbitrarily set at 25 after some speculative trials. Both models build their dictionary from the lyrics provided and then use the bag of words model to process queries. Part of speech is not part of this model.

For the SVM, the model was built using Sklearn library. The model trains using tf-idf vectorization of the corpus. This model only returns recall @ 1, but also can alternate part of speech counts.

Naïve Bayes was built by our team, and runs on word counts. No add one smoothing occurs, and there is no smoothing normalization in the denominator by the size of the total vocab, because we believed that there was importance in the individual lengths

of documents in the artist classes.

Locality Sensitive Hashing is built on the Datasketch library. It has limited functional performance. The model is built on token shingles of size one.

Decision Tree model was built on bag of words and tf-idf, as well as both text and the part of speech counts. The actual model is built from the sklearn library.

The vector space – cosine similarity model was built with tf-idf vectors. The model was built by the team and consists of the popularly defined tf-idf and cosine similarity formulas.

8. Results

Our results, shown in the tables and charts below, depict the recall of the models based on certain defining characteristics in how their individual experiments were run.

Classifier Recall @ K by Model [Optimal Feature Configuration]			
Model:	**Recall @ 1**	**Recall @ 5**	**Recall @ 10**
Decision Tree	0.3214	0.5119	0.7166
Naive Bayes	0.1595	0.4690	0.7262
LSI	0.1928	0.4714	0.6405
LDA	0.1309	0.3952	0.5738
Cosine Similarity	0.0338	0.1402	0.2447
LSH	0.0000	0.1000	0.1737
SVM	0.1119	(not available)	(not available)

Table 1 (above): *This table demonstrates recall of the models when tested on full length songs, and in their optimal configuration of preprocessing features.*

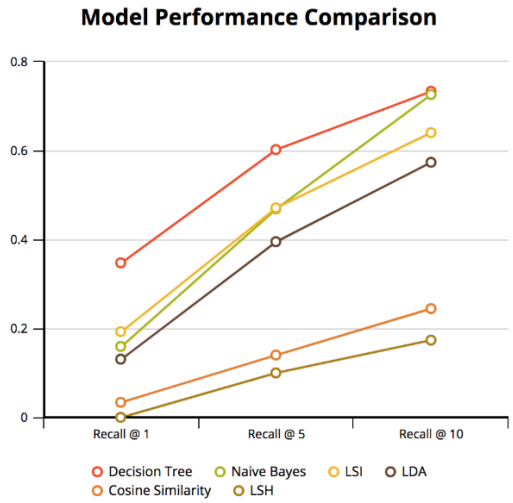

Figure 1 (above): *Graphical representation of Table 1*

Classifier Recall @ K by Model [Song Snippet]			
Model:	**Recall @ 1**	**Recall @ 5**	**Recall @ 10**
Decision Tree	0.316	0.483	0.688
Naive Bayes	0.141	0.47	0.707
LSI	0.1547	0.4071	0.5666
LDA	0.0547	0.25	0.4023
Cosine Similarity	0.0315	0.1105	0.2052
LSH	0	0.08	0.09

Table 2 (above): *This table demonstrates recall of the models when tested on shortened songs, and models in their optimal configuration of preprocessing features.*

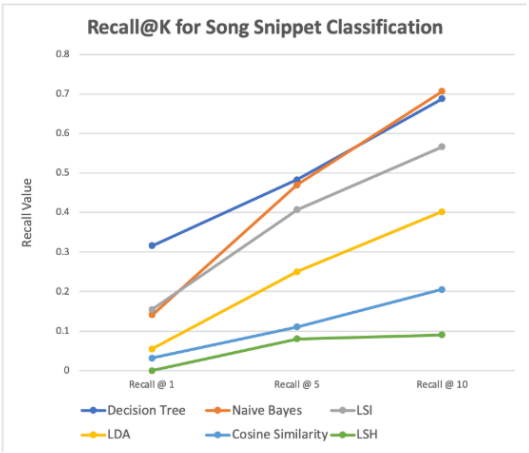

Figure 2 (above) : *Graphical Representation of Table 2*

Classifier Recall@1 and Feature Usage				
Model	**Base**	**Stopwords**	**Rap SW**	**Lemma**
Decision Tree	0.3357142857	0.3285714286	0.3238095238	0.3357142857
Naive Bayes	0.1119047619	0.1452380952	0.1142857143	0.1095238095
LSI	0.1785	0.180952381	0.1595238095	0.1928571429
LDA	0.1119047619	0.06904761905	0.09285714286	0.08333333333

Table 3: *Table showing the effect of inclusion of a preprocessing feature on the model recall performance.*

9. Conclusions/Implications

There are many promising and interesting results arising from this experiment. Overall, we attempted to determine how various types of classifiers and various configurations of preprocessing steps and feature usage would affect the recall performance of the models.

The results (Table 1) showed that decision trees were the optimal model type for this task. The recall @ 1 of around 32% very promising and demonstrates that strong classification may be possible with more data. Thereafter, Naive Bayes (NB) performed well, followed by LSI, and LDA. It's notable that even with simple word counts that NB was able to perform so well at nearly 16%, and demonstrates that even text alone is very potent. Finally, the topic modeling (LDA and LSI) demonstrate also that topics in rap yield enough differences for computers to identify. It is easy to take away from this that an combination of classifiers used in union may produce promising results.

It's important to note that the Cosine Similarity and LSH did not produce adequate results (near guessing). This suggests that there may are limitations in text, and that these models may not be ideal for future study.

Table 2, also produced important results about classifier performance. The table demonstrates how size of input song effects classifier performance. In all cases, the models decreased from full length songs, which is expected. The loss of data however did not produce significant performance loss, especially for decision trees and naive bayes which is notable.

Table 3 demonstrates how individual features affect classifier performance. Though our methodology and introduction hypothesized that features are truly the most important part, it showed that further preprocessing has varying effects. No trends were seen with lemmatization, stopword removal, or rap stopword removal that suggested anything different beyond random variation. It is true that a combination of features does produce slightly more optimal results, but generally minimal preprocessing is a workable strategy.

Overall, it is shown that artist-lyric classification is very computable. Though our experiment lacked the rigor of a large corpus and exhaustive model comparison, there is significant trends in which models perform better, and that just generally researchers must study to find optimal preprocessing for each model.

Future work should definitely include state of the art NLP models like BERT, as well as generally powerful deep learning networks. With more data, more computational power, and more features there is significant promise for future work in this area.

Individual Contributions

Overall, our team worked effectively and efficiently to write the source code, web application and deliverable material. We met outside of class on a regular basis to assign tasks and set timely progress goals. Krishan managed the team and built a multitude of models. Brian was in charge of the completion of the written deliverables as well as consulting on the source code whenever there were errors, additional features or additional testing to be completed. Sawit built the web system, integrating it with the models, and demo production. Thomas served as the primary editor for all writing tasks, proofreading and formatting checkpoints, the final report, and the project presentation in addition to writing the code for the decision tree model. Vraj worked on communicating with genius API, collecting and formatting corpus data and naive bayes algorithm.

References

[1] Ryan, Patrick. "Rap overtakes rock as the most popular genre among music fans. Here's why."*USA Today,* 2018.

[2] Elayidom, Sudheep, et al. “Text Classification For Authorship Attribution Analysis.” *Advanced Computing: An International Journal (ACIJ ’13)*, 2013, pp. 1-10.

[3] Herd, Denise. “Changes in the Prevalence of Alcohol Use in Rap Song Lyrics, 1979-97.” *Addiction*, 2005, pp. 1258-1269.

[4] Herd, Denise. “Changes in Drug Use Prevalence in Rap Music Songs, 1979-97.” *Addiction Research and Theory*, 2009, pp. 167-180.

[5] Herd, Denise. “Changing Images of Violence in Rap Music Lyrics: 1979-97.” *Journal of Public Health Policy*, 2009, pp. 395-406.

[6] McNulty-Finn, Clara. "The Evolution of Rap." *Harvard Political Review.* 2014

[7] https://saisenberg.com/projects/lyrics-classifier.html

[8] https://towardsdatascience.com/natural-language-processing-and-rap-lyrics-c678e60073fb

[9] Helge Homburg, Ingo Mierswa, B¨ulent Moller, Katharina Morik and Michael Wurst. "A Benchmark Dataset For Audio Classification And Clustering."