Predicting Song Genre using Lyrics with LDA and Bert Model

Ke (Chloe) Liu, Xiaoquan Liu

1. Introduction

The same as natural language, lyrics are words that make up a song. Nevertheless, lyrics are not typically regarded as traditional natural language. Lyrics are composed of verses and choruses and can have an explicit or implicit meaning. Some lyrics are abstract and nearly incomprehensible. In these circumstances, their explanation places a strong emphasis on form, articulation, meter, and symmetry of expression [1]. Therefore, we would love to see how those current popular language models would be performed differently if we chose not to use traditional text but rather, using lyrics. In this project, we performed document classification on lyrics as a Natural Language Processing (NLP) problem, more specifically, as a topic modeling problem. We considered each of the song lyrics as a single document and the corresponding song genre as "tags" or classes. Our goal is to retrieve information from lyrics and predict the genre of a given song based on that information.

2. Data

We collected data from Mendeley Data, which provides a list of lyrics from 1950 to 2019 describing music metadata as sadness, danceability, loudness, acousticness, etc., and other information, including artist name, music genre, and so on [2]. We selected both genre and lyrics from the original dataset and did natural language processing to clean the data. We deleted all non-English words in the lyrics, removed punctuation, and double spaces, and changed all words to lowercase in order to better encode them in our models. Since words can be grouped together based on their lemma and be analyzed as a single item or token, we lemmatized words in lyrics using WordNet's built-in Morphy function. In addition, words such as 'the', 'a', 'in' provide no useful information in lyric classification. Therefore, we used 'stop words' packages from the NLTK library to remove all stopwords in English. After data pre-processing, our final dataset contains 28,031 lyrics in seven genres: blues, country, hip hop, jazz, pop, reggae, and rock. For model training and testing, we split the training and testing data sets in an 80/20 ratio.

3. Model

3.1 Generative Model

We chose Latent Dirichlet Allocation (LDA) as our generative model and used the Latent Dirichlet Allocation with the online variational Bayes algorithm package from scikit-learn [3][4] to classify seven music genres. Since LDA is a type of unsupervised machine learning, we can only specify how many topics we want to find, which is seven. The seven classified topics that were given by the model were not necessarily the same as our original genres. After fitting the data into the model, we printed the top 50 words in each topic and used the pyLDAvis package to visualize the model results [5]. Table 1 shows the top 50 words in each topic. As shown in Figure 1, there are some overlaps between Topic1 and Topic 2. Since for each document there is a topic distribution, we then used the argmax function on each document's topic distribution to assign each document to a topic. In order to map the topics that were modeled by LDA to our original genres, we found the top 50 words in each original genre, computed the number of overlap words between each genre and each topic, and mapped each topic to the genre with the most overlapping words.

Table 1: *Top 50 Words in Each Topic*

Topic	Top 50 Words
0	world, like, bitch, time, real, high, smoke, fuck, people, yeah, know, life, need, face, live, cause, come, hell, roll, money, look, place, tell, city, stay, sell, think, go, watch, teach, right, work, shit, fake, young, ghost, deal, line, ride, stick, rule, paper, read, straight, race, drug, drink, mind, school, feel
1	heart, baby, go, hold, know, long, night, tonight, leave, right, believe, like, time, want, kiss, stay, come, feel, need, gonna, break, eye, wait, tear, tell, cause, arm, girl, sweet, hand, close, little, yeah, darling, love, start, touch, look, promise, apart, say, wrong, wanna, true, think, dream, fall, tight, light, line
2	away, good, break, walk, know, lonely, feel, night, go, wish, fool, miss, stand, come, dream, leave, goodbye, gonna, yeah, tell, morning, heart, cry, sleep, little, look, whoa, cause, fade, rain, think, tear, say, like, take, baby, girl, time, memory, hurt, want, stay, blue, right, place, wonder, home, dear, hide, yesterday
3	fall, feel, come, eye, know, like, inside, head, cold, black, dead, blood, hand, stand, soul, burn, hear, fear, leave, pain, fight, face, save, lose, wall, kill, death, devil, lie, turn, tear, light, grind, die, wind, hide, follow, look, break, mind, word, heart, deep, hold, live, hell, speak, life, body, dream
4	like, yeah, know, cause, fuck, come, shit, better, wanna, want, money, gonna, tell, gotta, play, right, party, bout, shoot, think, girl, talk, stop, say, head, look, bitch, little, need, game, start, baby, feel, damn, good, kick, drop, get, beat, go, hand, everybody, watch, check, goin, work, black, pull, night, turn
5	sing, home, song, blue, hear, come, woman, play, music, right, sweet, remember, go, lord, bring, like, ring, long, hand, word, songs, listen, write, call, know, money, say, fight, land, sound, yeah, days, little, radio, night, people, band, time, tell, dance, guitar, summer, swing, head, christmas, kill, year, roll, soon, tune
6	time, life, live, know, change, mind, come, think, want, world, things, need, cause, dream, feel, like, lose, love, look, go, today, give, yeah, tell, days, hard, right, learn, better, leave, forget, true, gonna, wanna, hurt, somebody, take, tomorrow, turn, remember, make, long, reason, forever, inside, thing, ready, matter, easy, mean

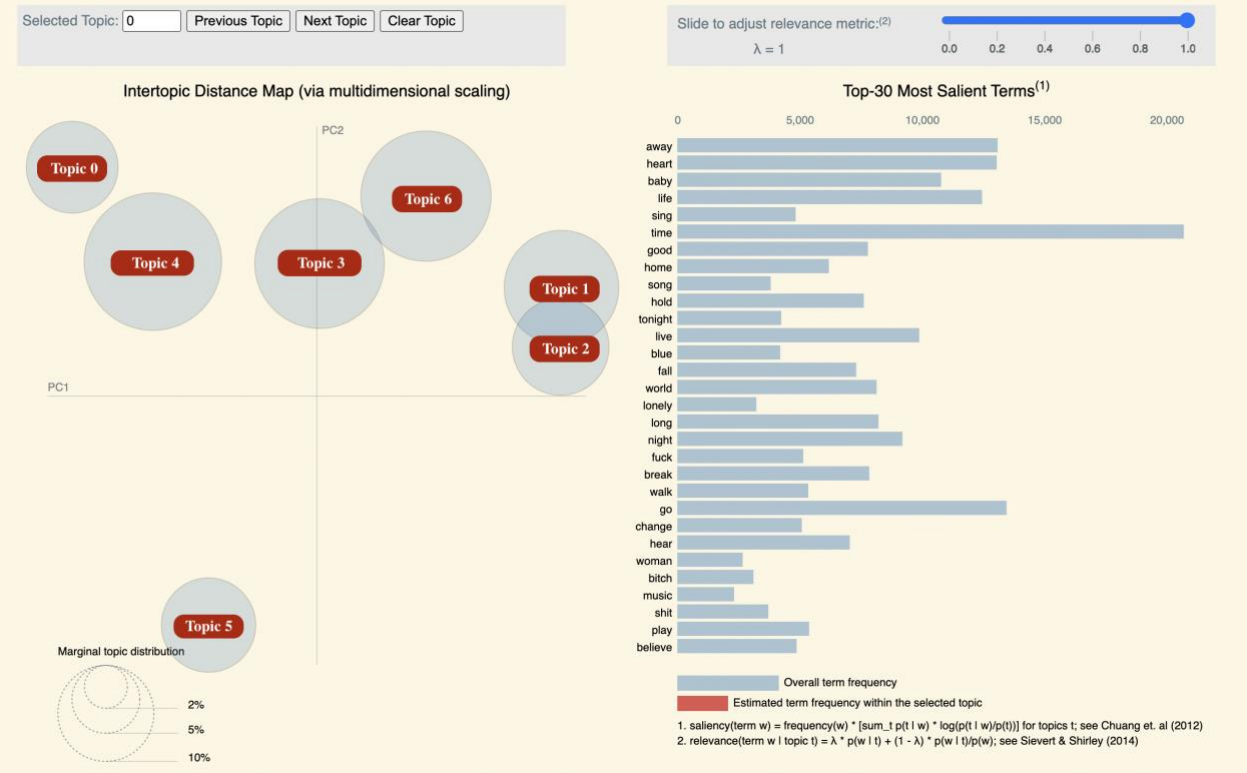

Figure 1: *Visualization using pyLDAvis*

Table 2 shows the mapping results of topics to genres as well as the result of applying our LDA model to real data, which achieved an overall correctness of 24.02% with relatively higher correctness on rock and hip-hop and relatively lower correctness on jazz and reggae.

Table 2: *Mapping Topics to Genre & Real Data Results*

Topic	Genre	Correctness
0	Jazz	0.0602
1	Pop	0.1849
2	Country	0.1361
3	Rock	0.3363
4	Hip-hop	0.6737
5	Reggae	0.0977
6	Blues	0.1926

Synthetic Data. We used the LDA model that we trained to generate synthetic data. For each genre, we generated 4,000 lyrics so that the size of the synthetic data is similar to the size of the real data. The length of each lyric is randomly chosen from the length of lyrics in the real data[6], and the words in each lyric are chosen based on the distribution of words for each topic.

Table 3 shows the samples of generated data for each genre. Table 4 shows the results of applying our LDA model to the synthetic data that is generated by itself. Since the generated data meets all the assumptions of our LDA model, our model achieved correctness above 99% across all genres.

Table 3: *Synthetic Lyrics for Each Genre*

Topic	Genre	Synthetic Lyrics
0	Jazz	blueprint sponsor daily see orange breed need jones world death bout unlikely scream play higher arguments abhorrent sense smoke dust hustle slide deal smoke high staffed struggle beast beat read world tick dodge people
1	Pop	tonight know hold lover free care stay crime long letter need long hold long break middle baby keep till heart long go kiss hold night darling heart right lover perfect ring heart sweet right stay little hold night hold mistake cause picture
2	Country	stocking news blow sleep cry away blue lovely leave break say dolls suddenly sweet tell cry piece turn lonely need farewell say come alright give
3	Rock	blind recover warm crack cruel world throne know roses mother greatest near come higher save funeral defect fall mother mouth idea line shape piece room doctor fight see shake feel shiny wire
4	Hip-hop	cause bout body make card fuck say gotta white know bitch everybody wanna gotta count beat buckle niki start shift right patch mama cold fuck tip sure noise fireworks turn better music clean thing ashamed hear wild seat everyday bloody
5	Reggae	sweet home good home come smile belong sing balloon go boys bloody kings music promise dine kill friend river brother little high look inspiration woman understand home sweet happen hand
6	Blues	hug break kind hide life everyday go keep shame know single time life late feel free thoughts destination world life believe gonna better room thing hate loan sure lose yeah outside dice time world dream honestly weasel mean better live world

Table 4: *Synthetic Data Results*

Topic	Genre	Correctness
0	Jazz	0.9997
1	Pop	0.9997
2	Country	0.9991
3	Rock	0.9994
4	Hip-hop	0.9991
5	Reggae	0.9997
6	Blues	0.9997

3.2 Discriminative Model

We developed the discriminative neuro network classifier by fine-tuning BERT (Bidirectional Encoder Representations from Transformers) model, which is a pre-trained model on the English language using a masked language modeling (MLM) objective. Compared with other language models, BERT's key technical innovation is applying bidirectional training by jointly conditioning on both left and right context in all layers to pre-train deep bidirectional representations from unlabeled text [7]. BERT adopted Transformer, a popular attention model, as its attention

mechanism to learn the contextual relationships between words (or subwords) in a text in a self-supervised manner.

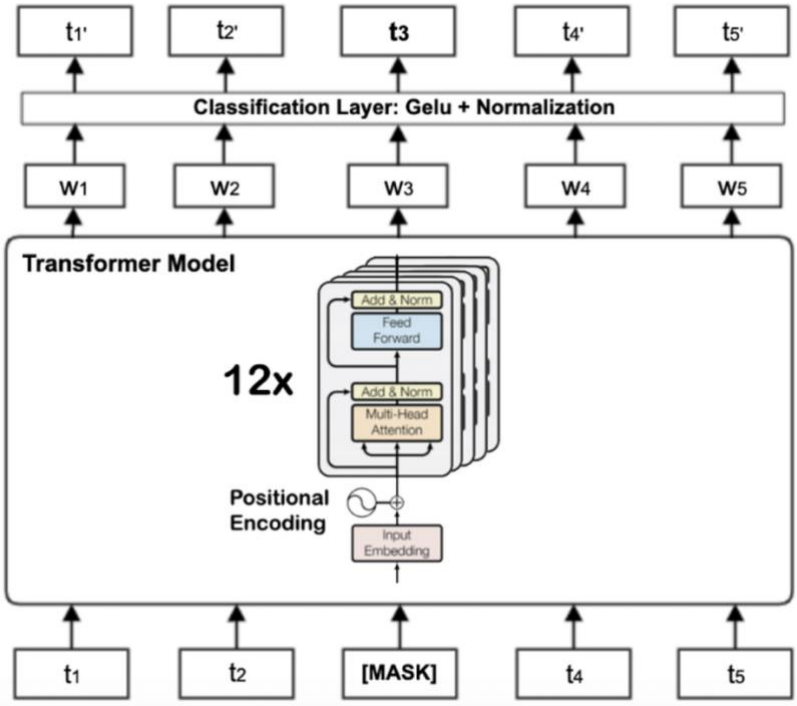

Figure 2. The Transformer based BERT base architecture with twelve encoder blocks [8]

Figure 2 is a high-level description of BERT base architecture. Based on Devlin's paper [7], BERT base model has 12 layers (Transformer blocks, denoted with L), a hidden size (denoted with H) of 768, and 12 self-attention heads (denoted with A), with total parameters of 110M. The input of BERT base model is a sequence of tokens, first embedded into vectors and then processed in the neural network. The output is a sequence of vectors of size H, in which each vector corresponds to an input token with the same index. To achieve context learning for bidirectional usage, BERT uses two training strategies: Masked LM (MLM) and Next Sentence Prediction (NSP).

Masked LM (MLM). Before feeding word sequences into the BERT, 15% of each sequence's words are swapped out for [MASK] tokens. Each word in a sequence is represented by a token in the masked word retrieval method. Based on the context of the words, this information is used to predict their value. Technically, for the prediction of the output words, the neuro network needs to add a classification layer on top of the encoder output, multiply the output vectors by the embedding matrix to convert them into the vocabulary dimension, and compute the probability of each word in the vocabulary using SoftMax. The prediction of the non-masked words is disregarded by the BERT loss function, which only considers the prediction of the masked values. The increased context compensates for the model's slower convergence rate compared to directed models.

Next Sentence Prediction (NSP). In the BERT training phase, the model learns to predict whether the second sentence in a pair will come after another in the original document by receiving pairs

of sentences as input. During training, 50% of the input pairs have a second sentence that is the next sentence in the original text, while in the other 50%, the second sentence is a randomly selected sentence from the corpus. The underlying presumption is that the second phrase will not be connected to the first. Before entering the model, the input is handled in the following manner to assist the model in differentiating between the two sentences during training. The first sentence has a [CLS] token at the start, and each subsequent sentence has a [SEP] token at the end. Each token has a sentence embedding that designates Sentence A or Sentence B. Token embeddings with a vocabulary of 2 and sentence embeddings share a similar notion. Each token receives a positional embedding to denote its place in the sequence.

The actions below are carried out to determine whether the second statement is related to the first: The Transformer model processes all of the supplied data. A straightforward classification layer is used to convert the [CLS] token's output into a 2*1 shaped vector (learned matrices of weights and biases). use SoftMax to determine the likelihood of IsNextSequence.

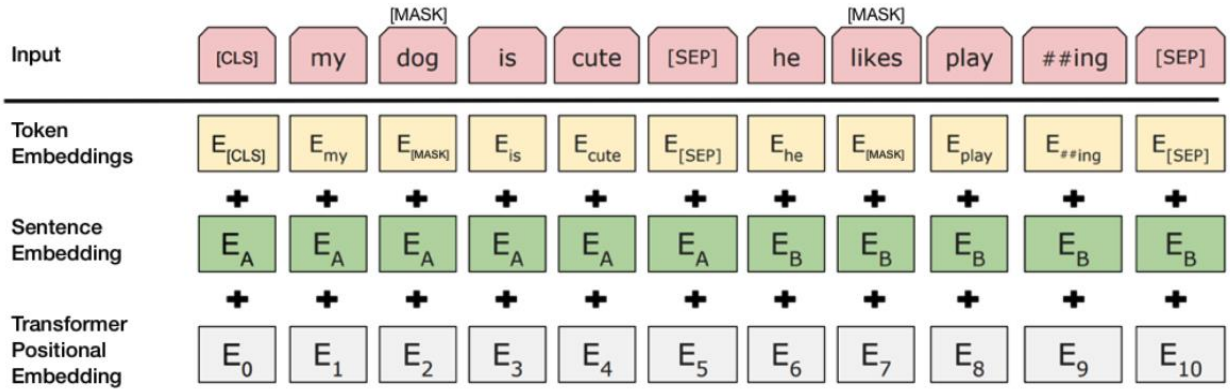

Figure 3. Next Sentence Prediction (NSP) [9]

When training the BERT model, Masked LM and Next Sentence Prediction are trained together, with the goal of minimizing the combined loss function of the two strategies. We used the training data to optimize model parameters while comparing the loss and accuracy on the validation model to optimize the hyperparameters. We apply cross-entropy loss as the loss function to the final Softmax layer in the model to generate our final topic classifications. Learning rate is one such hyper-parameter that defines the adjustment in the weights of our network with respect to the loss gradient descent. It determines how fast or slow we will move toward the optimal weights. We initialized the learning rate with 1e-1. The Gradient Descent Algorithm estimates the weights of the model in many iterations by minimizing a cost function at every step [10].

Model Training. Specifically, we utilized the BERT base model uncased model, that is, not case-sensitive for inputs, for the lyrics genre classification problem [11]. We first tokenized our input texts using BertTokenizer from the Hugging-face tokenizer library, with an initialized max_length of 256 and produced indices of the input tokens from the vocab file and attention masks for sequence classification. For output, we initialized it with an empty array with the dimension of length training set * number of unique classes (28031*7). We then used one-hot encoding to encode the target tensor. To load data easily, we created a data pipeline using TensorFlow dataset utility, to create batches of data. We then defined a function named LyricsDatasetMapFunction to convert input data to the required format with input_ids and attention masks. We initialized our

data to 16 batches and split our data into training and test sets with a ratio of 80:20. That is, for each of the 16 batches of data, we will have len(df)//16 samples, take 80% of that for training.

We initialized our model with a bert-based uncased model with pre-trained weights and built a frame around BERT using the typical tf.keras layers. There are a few parts to this frame: two input layers (one for the input IDs and another for the attention mask); a post-BERT dropout layer to reduce the likelihood of overfitting and improve generalization; a max pooling layer to convert the 3D tensor output by BERT, to a 2D tensor; and final output activations using softmax so that we can get our winning class prediction. We then feed in the input/output layers to the initialization function.

In terms of the training process, we trained the neuro network model with both original lyrics data synthetic data generated by LDA model. We first compile the model with suitable training parameters we were using an Adam optimizer with learning rate of 1e-5 and weighted decay of 1e-6. For our loss and accuracy, we employed categorical cross-entropy and categorical accuracy, respectively. We use categorical metrics here due to the fact that our one-hot encoded multi-class output labels are categorical. We then begin training with model.fit with epochs = 5, which means the entire data set has to be worked five times through the learning algorithm. Our model gets an accuracy near 70% after just five epochs.

4. Discussion

The "bag of-words" assumption of the LDA model lost information about the order of the words when trying to classify lyrics. Also, the LDA model performs worse at capturing rare topics due to data quality issues. Due to an increase in observations, our LDA model can be improved by training with a larger dataset. With the LDA, documents are represented as mixtures over latent topics, but when we assigned topic labels to lyrics, we simplified by choosing the topic with the highest probability, and mapping the pre-annotated genre of each lyric to a topic based on the number of overlapping words. A more reasonable approach can be taken to further improve the model's performance. During the process of generating data, since we randomly chose words based on the distribution of words for each topic, as we can see from Table 3, the synthesized lyrics are simply accumulations of words instead of meaningful sentences.

Table 5 shows the result of the two models on both the real and synthetic data. Compared to our discriminative model, the LDA model performed worse on the real data but better on the synthetic data that was generated by itself.

Table 5: *Overall Correctness*

Model	**Correctness (Real Data)**	**Correctness (Synthetic Data)**
LDA	0.2402	0.9995
BERT	0.6998	0.9994

The overall BERT model trained on real data has an accuracy not as high as models that are fine-tuned with traditional natural language but still performed better than the LDA model. Surprisingly, the BERT model performs well when training with synthetic data. When cross-checking with real

and synthetic data (Table 6), i.e., using synthetic data to check the model fine-tuned with real data and using real data to check the model fine-tuned with synthetic data, the accuracy for both of them is rather low. Since the BERT model will take context information into account, for the model on real data, when feeding in synthetic data generated by the LDA model, it cannot make enough sense of those distinct words that are not coherent into real sentences. Meanwhile, for the model was fine-tuned with synthetic data, it lost the context information at the training stage, then the prediction on real data will correspondingly be worse.

Table 6: *Bert Correctness by Model and Class*

Genre	**Correctness (Model on Real Data)**	**Correctness (Model on Synthetic data)**
Overall	0.166	0.132
Blues	0.002	0.000438
Country	0.000	0.000000
Hip Hop	0.000	0.000000
Jazz	0.447	0.630285
Pop	0.006	0.000289
Reggae	0.626	0.019481
Rock	0.084	0.318591

Even though the BERT model is an innovative breakthrough in language modeling, the main drawbacks of using BERT and other big neural language models are the computational resources needed to train/fine-tune and make inferences. For the BERT model, it takes 34 mins to run on GPU and more than one and a half hours to run on the CPU. Also, we didn't perform Grid Search on the BERT model due to computational constraints, which means we haven't trained our hyperparameter to the optimized level. In terms of interpretability, deep learning, Transformer based models can understand non-linear relations in the data and often have high accuracy but low interpretability due to their complex structure. As for the generative model, a big advantage of the LDA model is that it has fewer computational requirements because it can be trained locally within 10 seconds. With the algorithm and related distribution of an LDA model, it is also more interpretable.

Reference

[1] Wikipedia Contributors, “Lyrics,” Wikipedia, Jan. 05, 2020. https://en.wikipedia.org/wiki/Lyrics

[2] L. Moura, E. Fontelles, V. Sampaio, and M. França, “Music Dataset: Lyrics and Metadata from 1950 to 2019,” *data.mendeley.com*, vol. 2, Aug. 2020, doi: 10.17632/3t9vbwxgr5.2.

[3] “blei-lab/onlineldavb,” *GitHub*, May 17, 2022. https://github.com/blei-lab/onlineldavb

[4] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, “Stochastic Variational Inference,” *Journal of Machine Learning Research*, vol. 14, no. 4, pp. 1303–1347, 2013, Accessed: May 18, 2022. [Online]. Available: https://jmlr.org/papers/v14/hoffman13a.html

[5] B. Mabey, “bmabey/pyLDAvis,” *GitHub*, Oct. 15, 2020. https://github.com/bmabey/pyLDAvis

[6] A. Dai, “Duke NLP Final Project,” *GitHub*, Jan. 02, 2022. https://github.com/dai-anna/Duke-NLP-FinalProject

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. arXiv, 2018.

[8] Khalid, Usama & Beg, Mirza & Arshad, Muhammad. (2021). “RUBERT: A Bilingual Roman Urdu BERT Using Cross Lingual Transfer Learning.”

[9] Horev, R. (2018, November 17). “Bert explained: State of the art language model for NLP.” Retrieved December 18, 2022, from https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270

[10] Rakhecha, A. (2019, July 07). “Understanding learning rate”. Retrieved December 18, 2022, from https://towardsdatascience.com/https-medium-com-dashingaditya-rakhecha-understanding-learning-rate-dd5da26bb6de

[11] “theartificialguy”, “NLP-with-Deep-Learning,” *GitHub*, Nov. 30, 2021. https://github.com/theartificialguy/NLP-with-Deep-Learning/blob/master/BERT/Multi-Class%20classification%20TF-BERT