Determining Song Similarity Using Deep Unsupervised Learning

Brad Ross, Prasanna Ramakrishnan

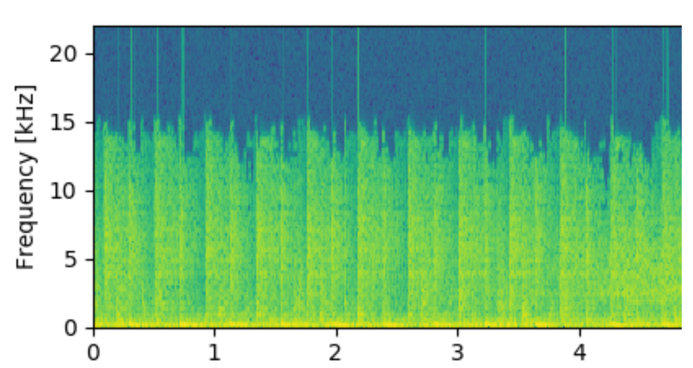
CS229, Fall 2017

Problem Setting

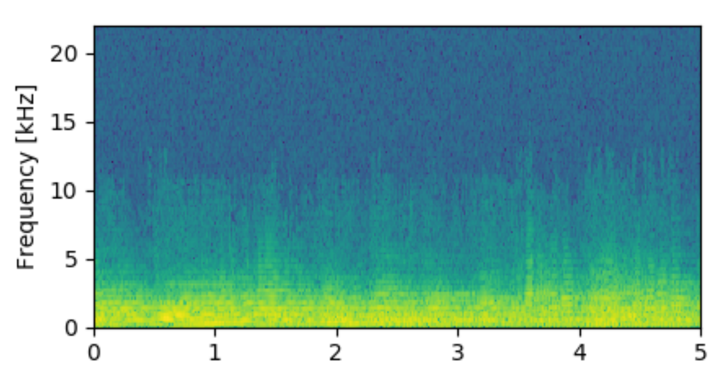
- **Goal**: Learn a metric space (M, d) and mapping function $f : S \to M$ (where S is the set of songs) such that $d(f(s_1), f(s_2))$ accurately captures the distance between songs s_1 and s_2.
- **Applications:** Fluid playlist creation, better song suggestions, geometric interpretations of genres and artists.

Dataset

- To generate our training dataset, we sampled ~1,000 30-second audio previews of songs from 15 different genres on Spotify.
- For each preview, we sampled ten 5-second segments to generate a total of ~150,000 training examples.
- To get a matrix representation of each example s, we computed $\text{spectrogram}(s) \in \mathbb{R}^{257\times 430}$ (using the short-time Fourier transform over windows).



Spectrogram of an EDM song



Spectrogram of a classical song

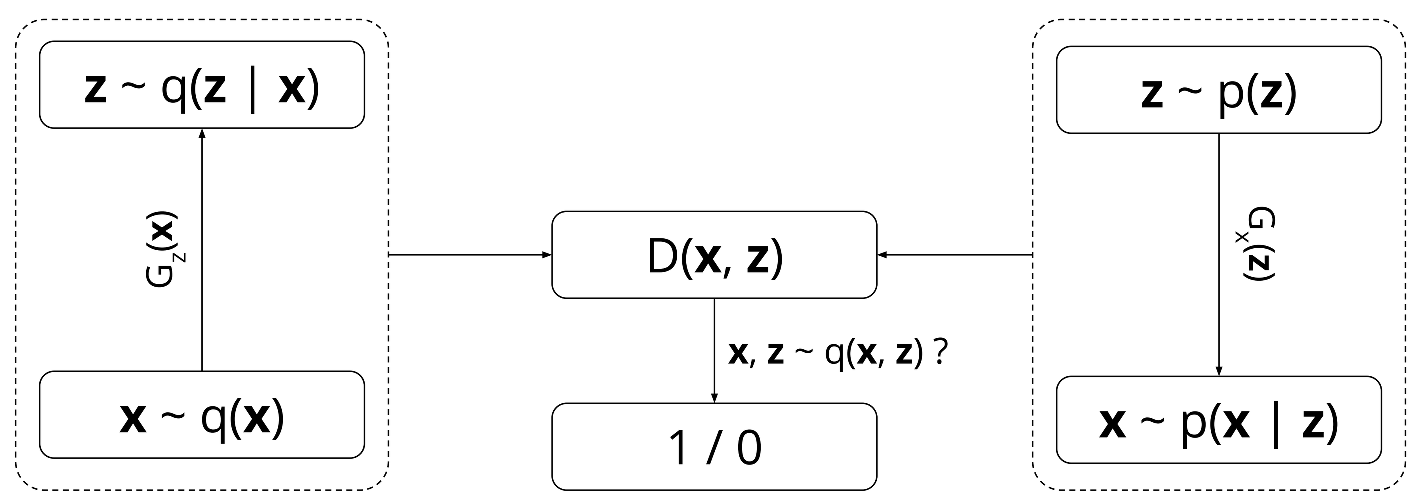
Embedding Techniques

Raw Embedding (baseline): Take $f_{\text{raw}}(s)$ to be the result of flattening the matrix $\text{spectrogram}(s)$. This gives us $f_{\text{raw}}(s) \in \mathbb{R}^{110510}$.

PCA embedding: Compute the r-component PCA of stacked raw embeddings to reduce the dimension. This gives us $f_{\text{PCA}}(s) \in \mathbb{R}^r$.

ALI embedding: Consider an adversarial game between three deep convolutional neural nets:

- An **encoder**, which maps spectrograms into vector
- A **decoder,** which maps vectors into spectrograms
- A **discriminator**, which given two examples, tries to guess which came from the encoder, and which came from the decoder.



Encoder **Discriminator** **Decoder**

The model is successful if the encoder and decoder are hard to distinguish, i.e., the encoder mapping is accurate, and invertible. We achieve this by optimizing the objective:

$$\max_{G_x, G_z} \min_{D} (\mathbb{E}_{x\sim q(x)}[D(x, G_z(x))^2] + \mathbb{E}_{z\sim p(z)}[(1 - D(G_x(z), z))^2])$$

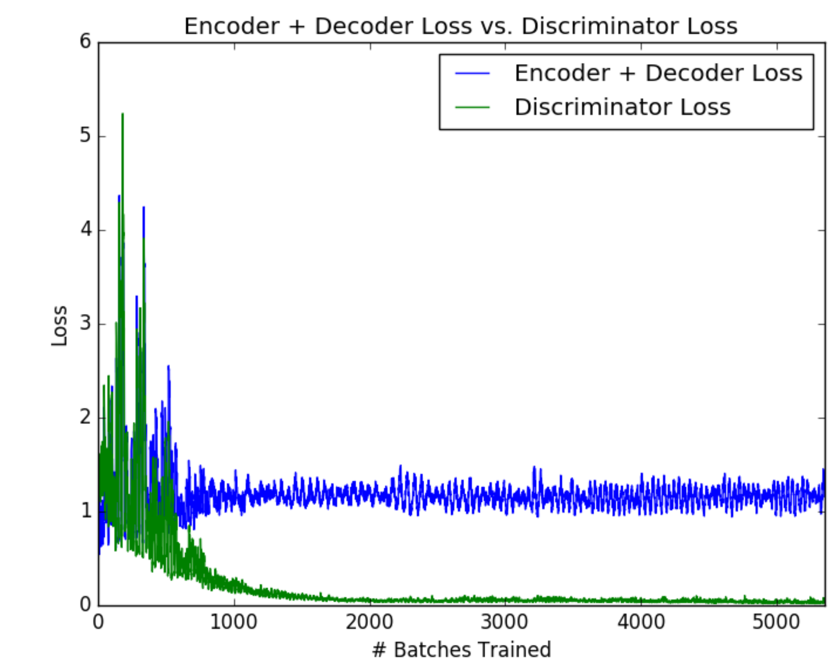
We use the output of the encoder for $f_{\text{ALI}}(s)$.

Evaluation Techniques

- We evaluate our embedding by how well **k-means** and **Mixture of Gaussians** perform on genre clustering.
- To evaluate clustering performance, we used the **V-measure** score, which measures how homogenous and complete a predicted clustering is, and **Adjusted Mutual Information** score, which measures agreement.

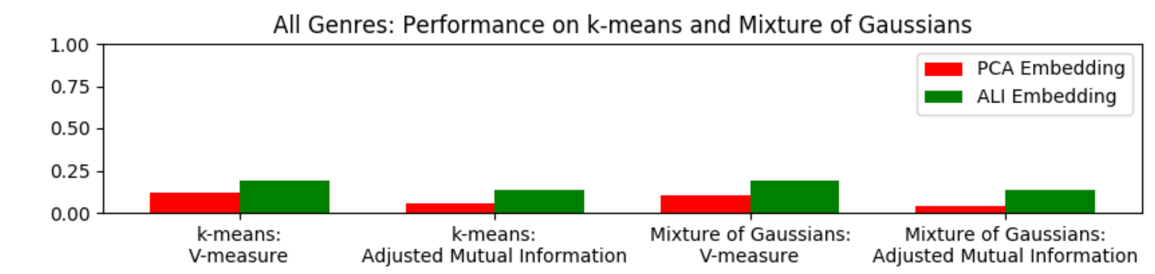
Results

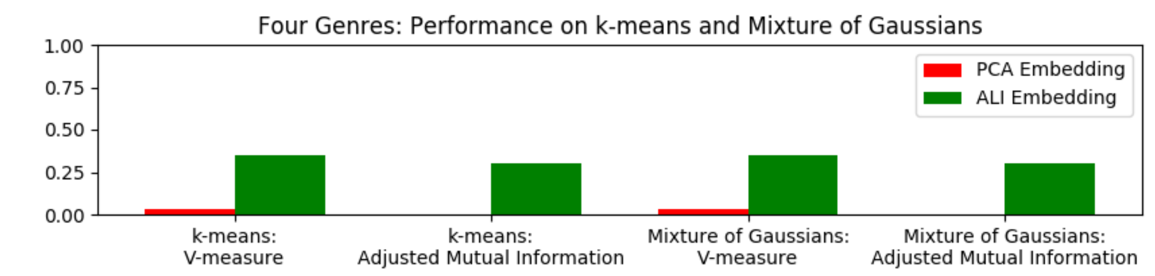
Learning



The encoder/decoder loss remains constant, despite the discriminator improving.

Performance

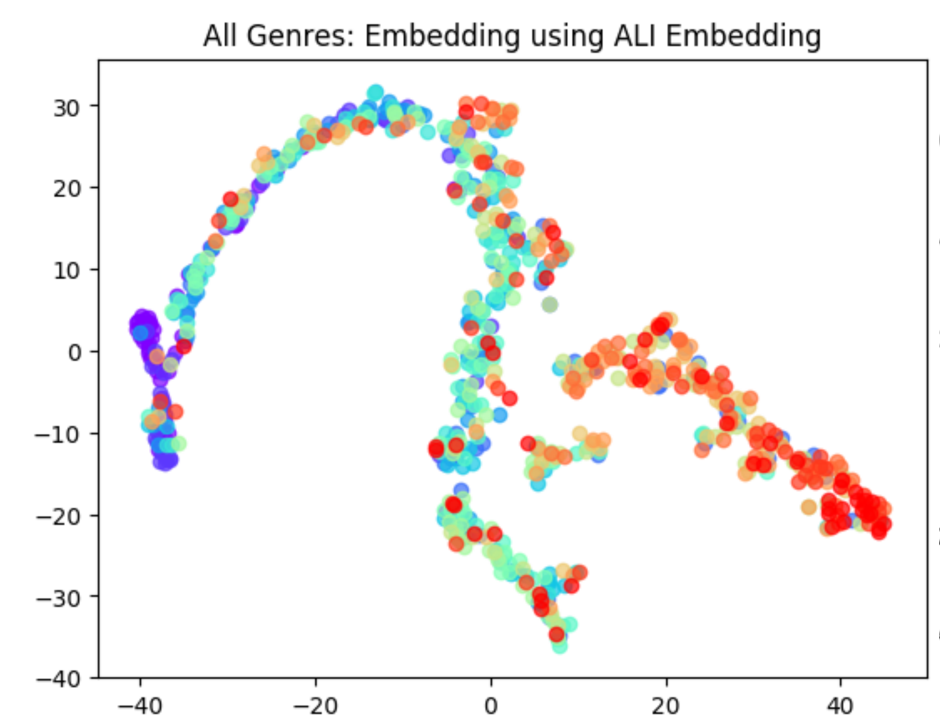


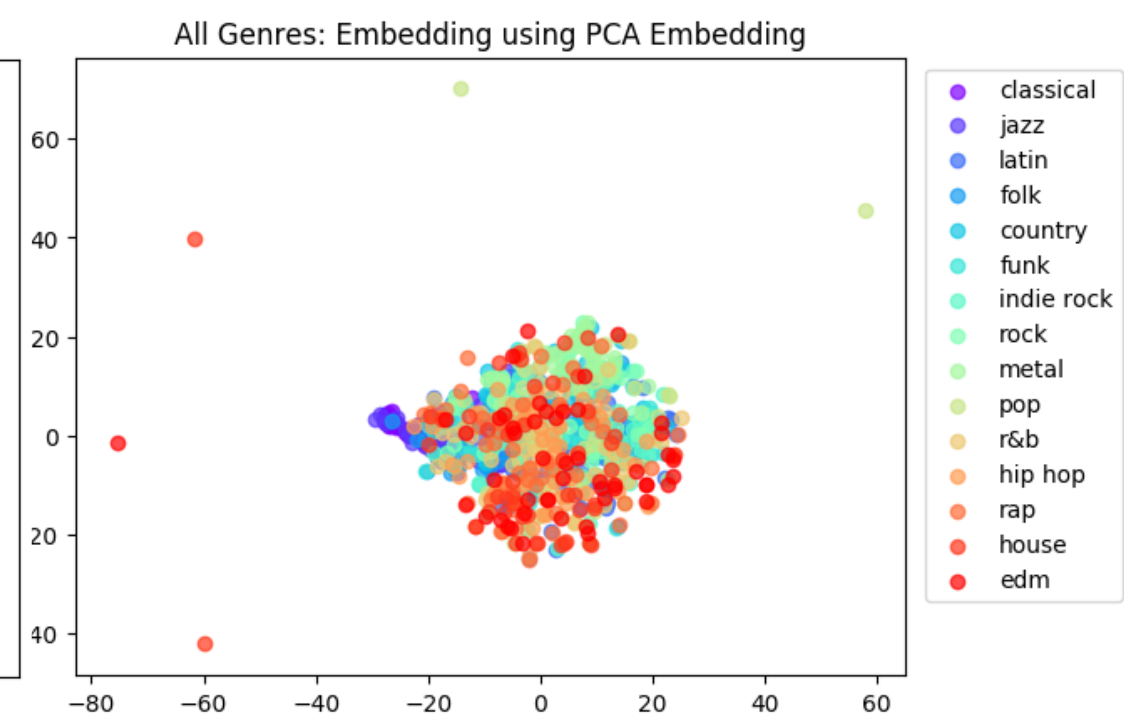


The ALI model outperforms the PCA model by a large margin

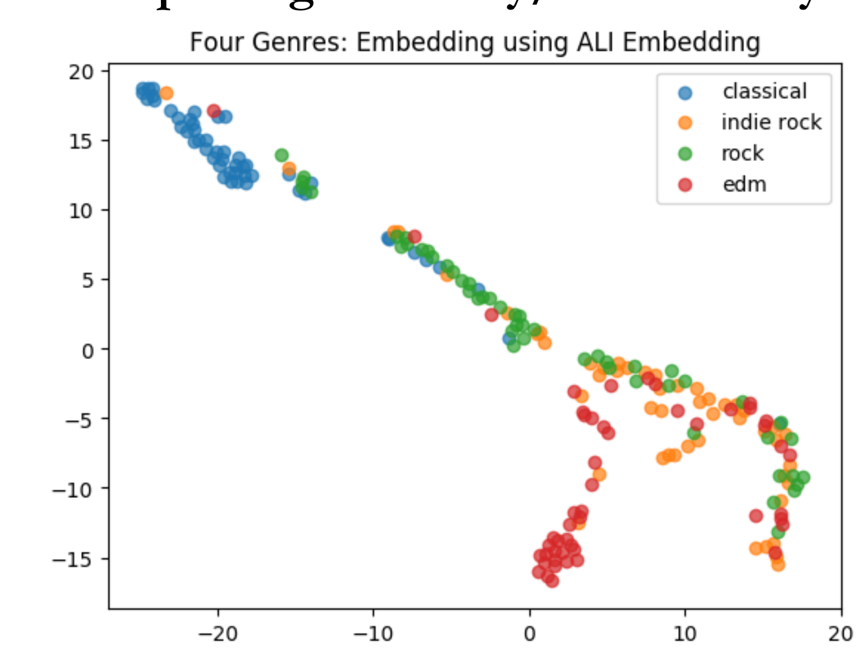
Visualizations

Embedding of 15 genres (50 songs each), with t-SNE Dimension Reduction

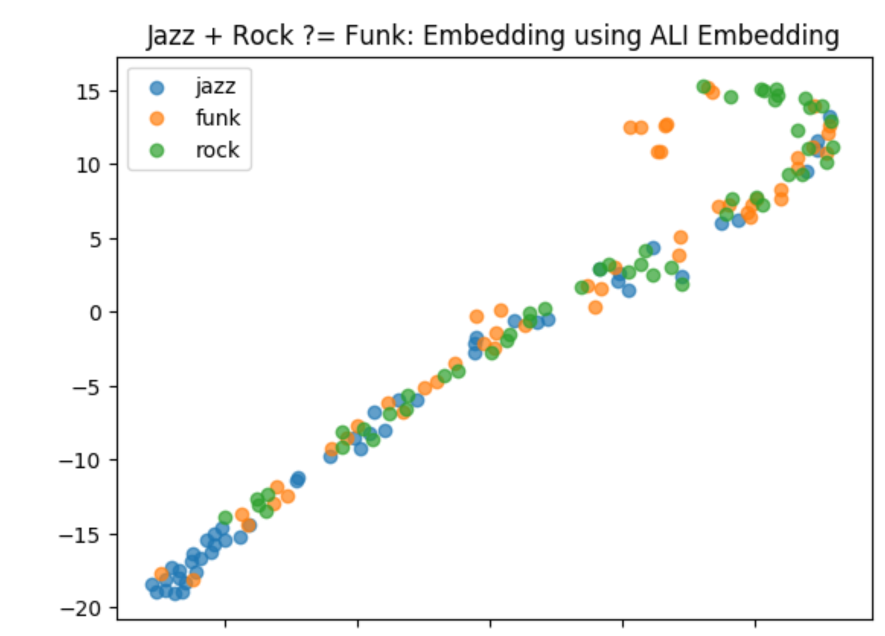




Capturing similarity/dissimilarity



Capturing Influences



See back of poster (next page) for citations

Bibliography

Chollet, F. (2015). Keras.

Dumoulin, V., Belghazi, I., Poole, B., Lamb, A., Arjovsky, M., Mastropietro, O., & Courville, A. (2016). Adversarially learned inference. *arXiv preprint arXiv:1606.00704*.

Maaten, L. V. D., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*(Nov), 2579-2605.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*(Oct), 2825-2830.