Music Creation through Non-negative Matrix Factorization

Jaskirat Kaur
Department of Computer Science
The University of Chicago
Chicago, IL
jaskiratk@uchicago.edu

Sarang Joshi
Department of Computer Science
The University of Chicago
Chicago, IL
sarangj@uchicago.edu

Trenton Wesley
Department of Computer Science
The University of Chicago
Chicago, IL
trentonjw@uchicago.edu

Abstract

In this paper, we explore non-negative matrix factorization (NMF) in the context of music samples. We show that NMF is a useful method for capturing the features of audio files. We also show that different loss functions in NMF algorithms can lead to a significant difference in the features captured. Finally, we successfully combine the features of different music samples to generate new music.

1 Introduction

Non-negative matrix factorization (NMF) refers to a group of linear algebra techniques that allows one to decompose a matrix into a product of two matrices. With NMF, a matrix, V is typically factorized into two matrices, W and H. Commonly, W is referred to as the feature matrix and H referred to as the coefficients matrix. All three of theses matrices have a non-negativity constraint that requires all elements to be greater than or equal to 0. This non-negativity constraint can be very useful for interpretability. Overall, NMF is a handy technique that has applications in bioinformatics [12, 7], computer vision [3], and astronomy [1, 13], among many others.

Previous work shows how NMF can be used to decompose music samples and derive representations for the musical components of the sample and their temporal arrangement [5, 8, 9]. NMF has been used in this way to remove certain sources of noise in music [6], to categorize music samples [9], and to create new music through the combining of features from different music samples [2, 11]. NMF has proved itself as a computationally inexpensive method for analyzing audio. However, there a number of NMF algorithms that have there various uses in different areas. These algorithms use different loss functions and can reach varying results. It remains an open question as to what algorithms work best regarding audio files.

In this paper, we set out to explore various NMF algorithms typically ultilzed in a range of applications. We will characterize these algorithms and there use in music decomposition, seeing how their results differ.

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

2 Related Work

2.1 Music Decomposition

The basic method in which people decompose these music samples takes two steps. In the first step, we take an music input signal and run a Short-Time Fourier Transform(STFT) on it. This input signal is some floating point time series data representing a music file. Running STFT, we get a complex-valued spectrogram matrix D from which we can decompose using NMF for our second step. We can then separate this matrix into its magnitude S and phase P components. In doing so, we get back two matrices W and H, where

$$S = WH$$

Here, if $S \in \mathbb{R}^{n \times p}$, then $W \in \mathbb{R}^{n \times k}$ and $H \in \mathbb{R}^{k \times p}$ for some $k > 0$. U would be our feature matrix, with each column u_i representing the individual musical components in our sample. W would represent the temporal arrangement of the musical elements in U. Multiplying these two matrices back together would give back a reconstructed version of our original matrix. From here, we can add our P phase components and run an inverse STFT to get back an audio sample to listen to.

2.2 Cost Function

We consider two cost functions to assess the quality of our approximation of the NMF Factorization $V \approx WH$. First we use Euclidean distance[10] which is widely used in measuring the dissimilarity between two matrices given by

$$\parallel A - B \parallel^2 = \sum_{ij} (A_{ij} - B_{ij})^2$$

The minimum value of this measure is zero and it only reaches zero when A exactly equals B. Another insightful matrix is defined as the Kullback-Liebler divergence:

$$D(A \parallel B) = \sum A_{ij} \log \frac{A_{ij}}{B_{ij}} - A_{ij} + B_{ij}$$

Similar to the Euclidean distance, the metric has a lower limit of zero and reaches zero only when A is precisely equal to B. However, it lacks symmetry in A and B leading us to term it the "divergence" of A from B. This matrix takes the form of the Kullback-Liebler divergence or relative entropy when $\sum A_{ij} = \sum B_{ij} = 1$.

Finally, as a reference, we perform the NMF of the sample matrix using the `scikit-learn` library's implementation [14] as a baseline.

2.3 Optimization for Music Decomposition

In our investigation, we address the music decomposition problem using non-negative matrix factorization (NMF), which is given by the following optimization problems:[10].

Problem 1: $\min_{W,H \geq 0} \|V - WH\|_F^2$
Problem 2: $\min_{W,H \geq 0} D(V \| WH)$

Note that while both the objective functions are convex in either W or H, they are not jointly convex in both variables. Consequently, finding a global optimum is not feasible. We can use gradient descent to approach the problem, however, it has a slow convergence. We are instead using multiplicative update rules to strike a balance between convergence rate and ease of implementation for NMF.

2.4 Multiplicative update rules

Theorem 1: *Under the iterative update rules, the euclidean distance*

$$H_{ia} \leftarrow H_{ia} \cdot \frac{(W^\top V)_{ia}}{(W^\top WH)_{ia}}, \quad W_{ia} \leftarrow W_{ia} \cdot \frac{(VH^\top)_{ia}}{(WHH^\top)_{ia}}$$

This distance remains invariant if and only if matrices W and H are at a stationary point with respect to it.

Theorem 2: *Under the iterative update rules, the divergence is non-increasing.*

$$H_{ia} \leftarrow H_{ia} \cdot \frac{\sum_k W_{ka} V_{ik}/(WH)_{ik}}{\sum_k W_{ka}}, \quad W_{ia} \leftarrow W_{ia} \cdot \frac{\sum_k H_{ka} V_{ik}/(WH)_{ik}}{\sum_k H_{ka}}$$

This divergence measure becomes stationary if and only if the matrices W and H have reached a stationary point with regard to this measure.

3 Methods

In order to compare the performance and results of the various objective functions in NMF, we chose two samples from the Ballroom dataset [4]: **I Like It 2** and **Latin Jam 2**, extracted 10-second samples, and performed the NMF-based musical decomposition with each of the three approaches discussed above: `scikit-learn`'s state-of-the-art approach, Euclidean distance, and divergence. Below is our implementation in Python of optimizing for the latter two objective functions:

```
def euclidean_nmf(V, n_components, n_iter = 1000):
  n, p = V.shape
  W = np.random.random((n, n_components))
  H = np.random.random((n_components, p))

  for i in range(n_iter):
    H_new = np.multiply(H, np.divide(W.T @ V, W.T @ W @ H))
    W_new = np.multiply(W, np.divide(V @ H.T, W @ H @ H.T))

    H = H_new
    W = W_new

  return W, H
```

```
def divergence_nmf(V, n_components, n_iter = 1000):
  n, p = X.shape
  W = np.random.random((n, n_components))
  H = np.random.random((n_components, p))

  for i in range(n_iter):
    W_H = W @ H
    H_new = np.zeros(H.shape)
    for a in range(H.shape[0]):
      for b in range(H.shape[1]):
        H_new[a,b] = H[a,b] * \
          np.sum(np.divide(np.multiply(W[:,a], V[:,b]), W_H[:,b])) / \
          np.sum(W[:,a])

    W_new = np.zeros(W.shape)
    for a in range(W.shape[0]):
      for b in range(W.shape[1]):
        W_new[a,b] = W[a,b] * \
          np.sum(np.divide(np.multiply(H[b],V[a]), W_H[a])) / \
          np.sum(H[b])

    H = H_new
    W = W_new

  return W, H
```

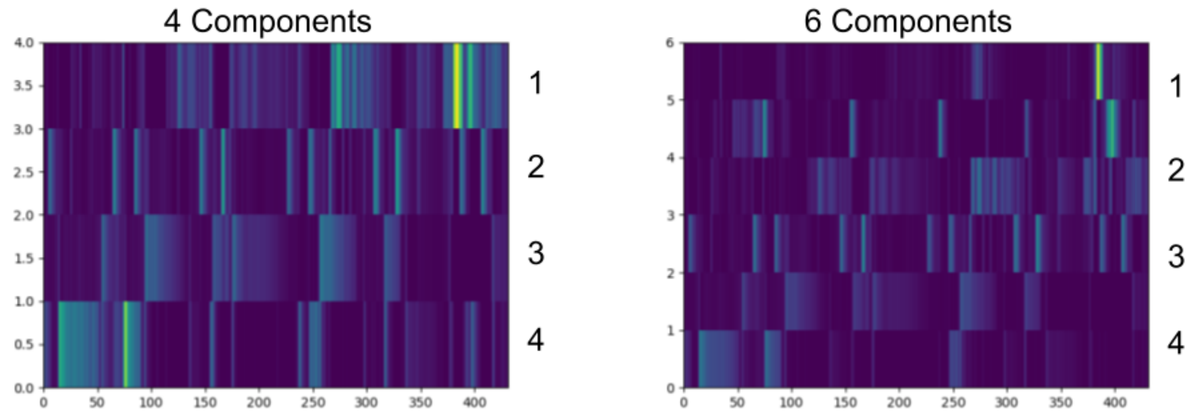

Figure 1: Shows H Matrices with 4 and 6 components respectively. Matrices were given by the use of the State-of-the-Art NMF decomposition method on **I Like It 2** audio file
.

First, we compared the output of running NMF with $n = 4$ and $n = 6$ components for the **I Like It 2** sample to contrast the component distribution of the two parameter values.

We chose $n = 4$ as the number of components in the two samples, based on the results of the previous comparison as well as a cursory auditory analysis of the range of instruments/frequencies in each.

4 Results & Discussion

4.1 Number of Components

We first observe the different H matrixes using the State-of-the-Art approach for **I Like It 2** with $n = 4$ and $n = 6$ components. We can see that the high value points of the components get translated/combined from $n = 6$ to $n = 4$. If we look at the concentrated high-value section in the top-right corner of our spectrograms, we can see that the $n = 4$ matrix's 4th row values have been spread across the 3rd and 4th rows of the $n = 6$ matrix at the same time sections.

As a result of this experiment, we decided to choose $n = 4$ as the number of components for the multi-sample comparison (in the next section). The components are more clearly defined with this value across the three approaches, and upon reconstruction of individual components, we see more clear differentiation with this value.

4.2 Feature Interpretation

The results of running the three NMF implementations are shown in the spectrograms for figures 2-5. For the W matrices, each column represents a component. For the H matrices, each row represents a component over the course of the 10-second sample. We can see from these figures that the different NMF approaches produced a different distribution of information across the 4 components.

In the State-of-the-Art approach for **Latin Jam 2**, we can see that the values of the 4th column has much higher values than the other three. This is somewhat similar to the Euclidean objective approach. However, in the divergence objective approach, the concentration of these high values is distributed much more evenly across the four columns.

Moving to the H matrices, which represent the temporal envelopes of the sample, we notice some strong similarities across the three approaches. We start with the graphs for **Latin Jam 2**. Take for

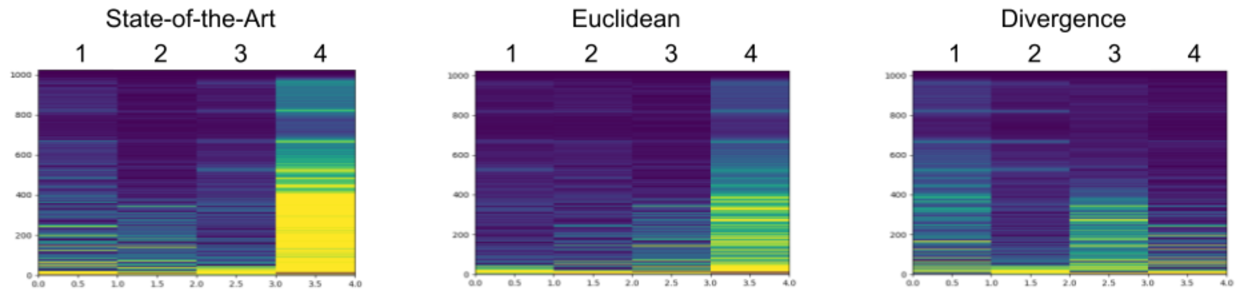

Figure 2: Shows W matrices for **Latin Jam 2**. Audio file is decomposed using the three compared methods: State-of-the-Art, Euclidean, and Divergence.

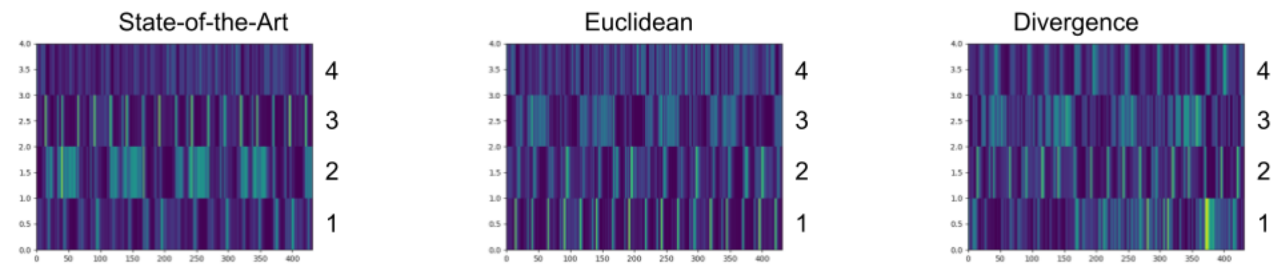

Figure 3: Shows H matrices for **Latin Jam 2**. Audio file is decomposed using the three compared methods: State-of-the-Art, Euclidean, and Divergence.

example the 3rd row in the State-of-the-Art approach. We see that there is a high value at a steady cadence, approximately every 30 columns. This appears to correspond to the steady beat that is in the music sample. Now if we look at the 1st row in the Euclidean graph and the 2nd column in the divergence graph, we see that the cadence is almost identical. The same similarity can be seen for **I Like It 2**: the 4th row in the State-of-the-Art approach loosely matches the 3rd row in the Euclidean approach and the divergence approach.

We can observe therefore that even though the exact nature of the individual components differs from approach to approach, all three are able to identify some fundamental musical aspects of the sample. The other components are also similar, with minor differences that are spread across other components.

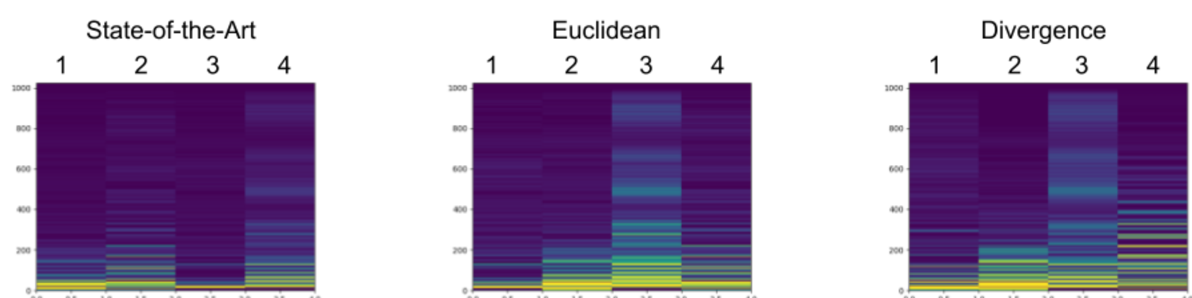

Figure 4: Shows W matrices for **I Like It 2**. Audio file is decomposed using the three compared methods: State-of-the-Art, Euclidean, and Divergence.

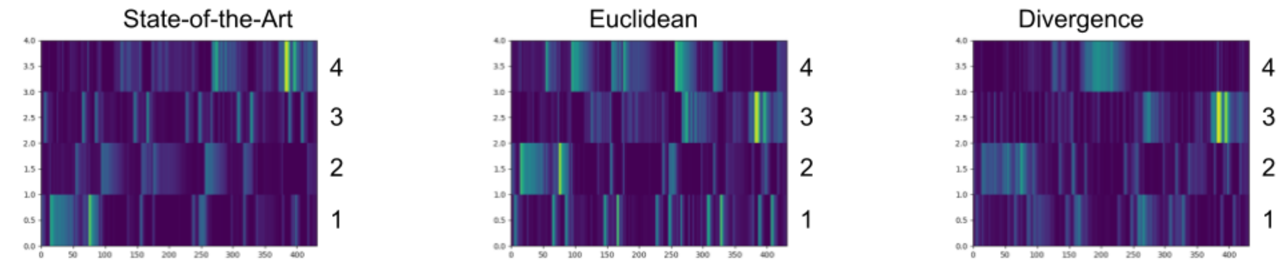

Figure 5: Shows H matrices for **I Like It 2**. Audio file is decomposed using the three compared methods: State-of-the-Art, Euclidean, and Divergence.

5 Conclusion

In conclusion, we have investigated the application of non-negative matrix factorization(NMF) for music decomposition. We explored different NMF algorithms, each with different loss functions, that exhibited distinct results when applied to audio files. By applying Euclidean distance and the Kullack-Leibler divergence as cost functions, we were able to characterize the performance of these algorithms in reconstructing and interpreting musical components from a given sample.

Through careful analysis of the W and H matrices for our chosen music samples, it became apparent that while some algorithms, particularly the state-of-the-art approach from scikit-learn, lean towards isolating certain musical instruments, others, particularly those driven by the divergence objective, provide a more evenly distributed representation across components.

We also experimented with the versatility of NMF in music-mosaic, that is creating new musical compositions by amalgamating features from different sources. Hence, NMF is not only an analysis tool but can also be used as a creative platform for musicians and engineers.

6 Additional Ideas

Our exploration raises question of algorithm selection based on the genre, complexity and desired outcome of music analysis. Our future work can include exploring custom-tailored NMF solutions that cater to specific artistic requirements thereby opening possibilities for the domain of automated music production.

References

[1] Olivier Berné, A Helens, Paolo Pilleri, and Christine Joblin. Non-negative matrix factorization pansharpening of hyperspectral data: An application to mid-infrared astronomy. In *2010 2nd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*, pages 1–4. IEEE, 2010.

[2] Andrew Boik and other. Improved audio mosaicing techniques for sound synthesis. Undergraduate senior thesis at Princeton University, 2013.

[3] Ioan Buciu and Ioannis Pitas. Application of non-negative and local non negative matrix factorization to facial expression recognition. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 1, pages 288–291. IEEE, 2004.

[4] Gouyon F., Klapuri A., Dixon S., Alonso M., Tzanetakis G., Uhle C., and Cano P. An experimental comparison of audio tempo induction algorithms. *IEEE Transactions on Speech and Audio Processing*, 14(5), 2006.

[5] Cédric Févotte, Nancy Bertin, and Jean-Louis Durrieu. Nonnegative matrix factorization with the itakura-saito divergence: With application to music analysis. *Neural computation*, 21(3):793–830, 2009.

[6] Derry Fitzgerald, Matt Cranitch, and Eugene Coyle. Shifted non-negative matrix factorisation for sound source separation. In *IEEE/SP 13th Workshop on Statistical Signal Processing, 2005*, pages 1132–1137. IEEE, 2005.

[7] Ryuji Hamamoto, Ken Takasawa, Hidenori Machino, Kazuma Kobayashi, Satoshi Takahashi, Amina Bolatkan, Norio Shinkai, Akira Sakai, Rina Aoyama, Masayoshi Yamada, et al. Application of non-negative matrix factorization in oncology: one approach for establishing precision medicine. *Briefings in Bioinformatics*, 23(4):bbac246, 2022.

[8] Marko Helen and Tuomas Virtanen. Separation of drums from polyphonic music using non-negative matrix factorization and support vector machine. In *2005 13th European Signal Processing Conference*, pages 1–4. IEEE, 2005.

[9] Andre Holzapfel and Yannis Stylianou. Musical genre classification using nonnegative matrix factorization-based features. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):424–434, 2008.

[10] Daniel Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems*, 13, 2000.

[11] Patricio López-Serrano, Christian Dittmar, Yigitcan Özer, and Meinard Müller. Nmf toolbox: Music processing applications of nonnegative matrix factorization. In *Proceedings of the 22nd International Conference on Digital Audio Effects (DAFx-19), Birmingham, UK*, pages 2–6, 2019.

[12] Ben Murrell, Thomas Weighill, Jan Buys, Robert Ketteringham, Sasha Moola, Gerdus Benade, Lise Du Buisson, Daniel Kaliski, Tristan Hands, and Konrad Scheffler. Non-negative matrix factorization for learning alignment-specific models of protein evolution. *PloS one*, 6(12):e28898, 2011.

[13] Ibrahim Selim, Arabi E Keshk, and Bassant M El Shourbugy. Galaxy image classification using non-negative matrix factorization. *Int. J. Comput. Appl*, 137(5):4–8, 2016.

[14] `https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.NMF.html`.